



## Vectorization of optically sectioned brain microvasculature: Learning aids completion of vascular graphs by connecting gaps and deleting open-ended segments

John P. Kaufhold<sup>a,\*</sup>, Philbert S. Tsai<sup>b</sup>, Pablo Blinder<sup>b</sup>, David Kleinfeld<sup>b,c</sup>

<sup>a</sup> SAIC, Arlington, VA, United States

<sup>b</sup> Department of Physics, UCSD, La Jolla, CA, United States

<sup>c</sup> Section of Neurobiology, UCSD, La Jolla, CA, United States

### ARTICLE INFO

#### Article history:

Received 21 July 2011

Received in revised form 1 June 2012

Accepted 8 June 2012

Available online 26 June 2012

#### Keywords:

Anatomy

Gap filling

Neurovascular coupling

Skeleton

Two photon microscopy

### ABSTRACT

A graph of tissue vasculature is an essential requirement to model the exchange of gasses and nutrients between the blood and cells in the brain. Such a graph is derived from a vectorized representation of anatomical data, provides a map of all vessels as vertices and segments, and may include the location of non-vascular components, such as neuronal and glial somata. Yet vectorized data sets typically contain erroneous gaps, spurious endpoints, and spuriously merged strands. Current methods to correct such defects only address the issue of connecting gaps and further require manual tuning of parameters in a high dimensional algorithm.

To address these shortcomings, we introduce a supervised machine learning method that (1) connects vessel gaps by “learned threshold relaxation”; (2) removes spurious segments by “learning to eliminate deletion candidate strands”; and (3) enforces consistency in the joint space of learned vascular graph corrections through “consistency learning.” Human operators are only required to label individual objects they recognize in a training set and are not burdened with tuning parameters. The supervised learning procedure examines the geometry and topology of features in the neighborhood of each vessel segment under consideration.

We demonstrate the effectiveness of these methods on four sets of microvascular data, each with  $>800^3$  voxels, obtained with all optical histology of mouse tissue and vectorization by state-of-the-art techniques in image segmentation. Through statistically validated sampling and analysis in terms of precision recall curves, we find that learning with bagged boosted decision trees reduces equal-error error rates for threshold relaxation by 5–21% and strand elimination performance by 18–57%. We benchmark generalization performance across datasets; while improvements vary between data sets, learning always leads to a useful reduction in error rates. Overall, learning is shown to more than halve the total error rate, and therefore, human time spent manually correcting such vectorizations.

© 2012 Elsevier B.V. All rights reserved.

## 1. Introduction

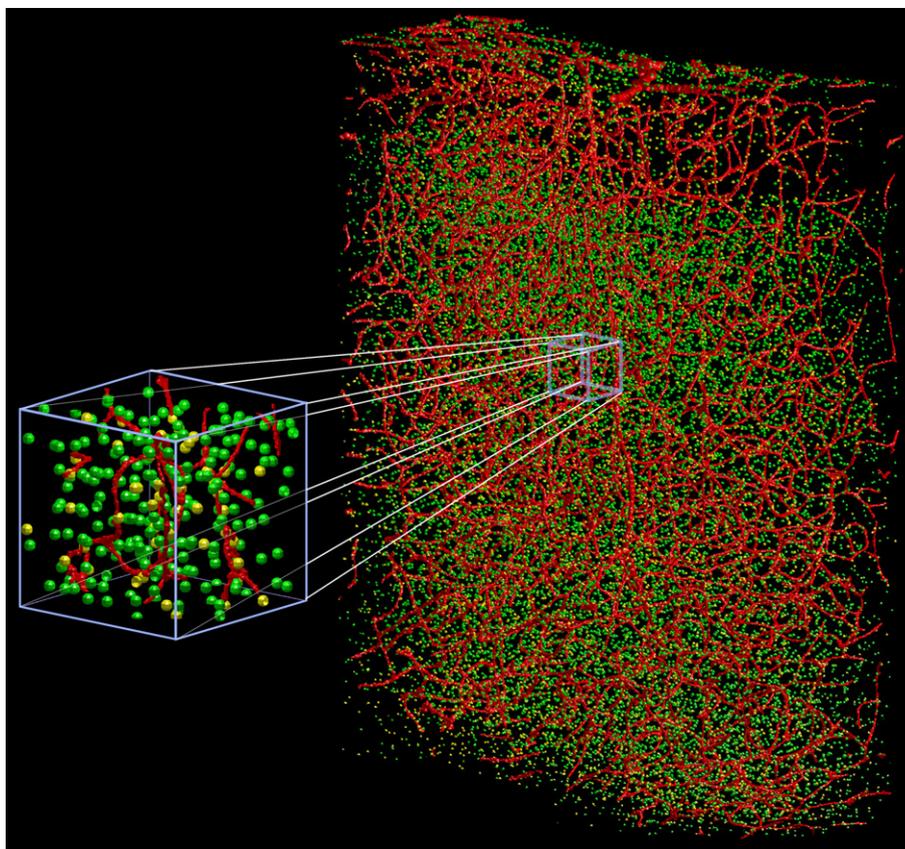
### 1.1. Background and motivation

The brain's ability to process information and coordinate sensory and motor function relies critically on the body's ability to deliver nutrients to the brain as well as to remove heat and waste from the brain's constituent neurons. In the mammalian brain, these functions are provided by the blood that flows throughout the brain through an interconnected three-dimensional (3D) vascular network (Fig. 1). A need to understand the topology and mor-

phology of the brain's vasculature has received renewed interest (Risser et al., 2007b; Heinzer et al., 2008; Cassot et al., 2006; Blinder et al., 2010) as a result of the clinical importance of blood flow in neurological disease states as well as the growing popularity of technologies that infer neuronal activity from blood flow. Positron emission tomography and magnetic resonance imaging are among the imaging modalities that exploit the neurovascular coupling between neurological activity, the ensuing oxygen and energy consumption, and changes to local blood perfusion to infer neuronal activity. Yet the microscopic details of the vascular network and its relationship to surrounding neuronal units remains poorly understood, and investigators continue to debate which specific aspects of the neuronal activity elicit these observable changes in blood flow (Devor et al., 2007; Logothetis et al., 2001). Furthermore, the spatial extent of the imaged response extends beyond

\* Corresponding author. Address: SAIC, 4001 N. Fairfax Drive, Ste. 400, Arlington, VA 22203, United States. Tel.: +1 703 469 3692 (O); fax: +1 703 248 7769.

E-mail address: [kaufholdj@saic.com](mailto:kaufholdj@saic.com) (J.P. Kaufhold).



**Fig. 1.** A thin vectorized section of microvasculature. Surface rendering of the vascular graph is shown in red, neuronal nuclei centroids are shown as green spheres, and non-neuronal nuclei are shown in yellow (abstracted from (Tsai et al., 2009b)). State of the art segmentation methods admit both gaps and spurious merges in vectorized representations of the microvascular network topology (red).

the anatomical limits of its corresponding neuronal origin (Grinvald et al., 2000; Fukuda et al., 2006; Schaffer et al., 2006), a phenomenon that is likely related to the topological properties of the vascular network.

It is necessary to generate complete, vectorized representations of the vascular connectivity over sufficiently large volumes so that edge effects are minimized, yet with sufficient resolution to capture the smallest microvascular elements. To accurately model the flow of blood in such a network, the graph should be free of both spurious connections and erroneous gaps. Although, in principle, direct error-free segmentation and vectorization of the vascular network should be possible, in practice there are several potential sources of error. These include upstream issues such as the completeness and stability of vascular staining, aberrations in the acquisition of images, and reduced signal-to-noise necessary to accommodate large-scale data collection in reasonable scan times. Furthermore, no segmentation method is guaranteed to be robust across all datasets and signal conditions. In general, the generation of topologically error-free vectorizations of 3D tubular structures from 3D grayscale data with tens of thousands of tubular segments is an unsolved problem. Specifically, the recent review of vascular segmentation by Lesage (Lesage et al., 2009), concludes: “Aiming for a generic, flawless segmentation framework is probably illusory”, which can be interpreted as follows: state of the art segmentation (even in the simpler vascular networks explored in all of the cited work in Lesage) will be incorrect in that it will connect some vessels spuriously and miss other connections that should be made between vessels. Related work in connectomics highlights the importance of scalable image segmentation algorithms that learn from labeled datasets (Jain et al.,

2010b; Kleinfeld et al., 2011). As one indication of the importance of such scalable automation, current state of the art in Serial Block Face Scanning Electron Microscopy (SBF-SEM) can acquire a cortical column in a year, but manually tracing the same volume with state of the art methods would take 1000 years (Helmstaedter et al., 2008). Though state of the art in connectomics in electron microscopy still relies heavily on manual tracing, newer work has improved labeling speed approximately 50-fold with a centerline skeleton representation (Helmstaedter et al., 2011). Consensus centerline skeletons from multiple labelers are then found by voting to either eliminate or keep centerline edges. This evaluation of many individual segments by voting lends itself to the kind of learning on object-level connections explored in this work—in both cases, some centerline strands are added and some are deleted based on voting or learning.

Importantly, the presence of even a few defects in the connectivity graph can result in significant errors in analytical results. Specifically, the flow of blood through a vessel scales approximately as the fourth power of the vessel radius. Because vessel radii can span an order of magnitude or more, the resulting flow can readily span several orders of magnitude. Therefore the erroneous addition or exclusion of even a single medium-sized vessel has the potential to either shunt flow from or induce flow into a significant volume of smaller vessels. Thus the detection and correction of even a small number of graph defects can significantly improve the quality of downstream analytical results. This corrected vascular graph forms the underlying model for calculation of blood flow in the brain. The presence of errors could lead to errors in estimates of vascular flow and concomitant perfusion of tissue with gases and metabolites.

Here we develop methods for the detection and characterization of graph errors in large vascular datasets, together with the application of machine learning to optimize and automate the correction of such errors. Our algorithms utilize a combination of topology-, morphometry- and intensity-based features to locate defects and to learn the defect correction decision. A consistency learning algorithm is developed to resolve cases in which learned decisions for strand removal and strand addition conflict. We build on previous work (Tsai and Kaufhold, 2009a; Tsai et al., 2009b) and image, segment, and vectorize labeled vasculature in cubic-millimeter-size blocks of brain tissue with micrometer resolution.

The paper is organized as follows: Section 1.2 introduces the types of errors the algorithms in this paper aim to correct and highlights the difficulties in the chain of sample preparation, acquisition and analysis that lead to these errors. Section 1.3 discusses published methods to address the problems described in Section 1.2. Section 2 introduces our Methods, with an overview of the relevant aspects of our acquired data in Section 2.1, followed by a discussion of the threshold relaxation method used to bridge gaps and a method to select candidate strands to delete from the vascular graph, respectively, in Sections 2.2 and 2.3. Section 2.4 introduces our labeled training data, features, bagged and boosted decision tree learning methods and our performance validation methods. Section 2.5 introduces the concept of consistency learning, which infers joint labels based on pairs of learned scores that interact with each other through common endpoints. Section 3 collects the performance results of our methods on real data, breaking down performance for each component of our method; bridging strand performance is addressed in Section 3.1, deletion of strands is addressed in Section 3.2, generalization error is addressed in Section 3.3, and consistency learning is addressed in Section 3.4. Section 4 discusses the key contributions of the paper; improvements to bridging strand performance is discussed in Section 4.1, the benefits of learning beyond the confidence metric are discussed in Sections 4.2 and 4.3, learning to delete strands is discussed in Section 4.4, consistency learning is discussed in Section 4.5, upstream acquisition and algorithm considerations are discussed in Section 4.6, and issues of generalization error are discussed in Section 4.7. Section 5 presents the overall conclusions and suggests the value of active learning as a next step.

## 1.2. Sources of vascular graph errors

The methods described in this paper take connectivity graphs, in the form of vectorized maps, as their starting point. The process of producing a vectorized graph from the topological structure of a real object can be divided into three stages: (1) data acquisition, (2) image segmentation, and (3) vectorization. Each stage has the potential to introduce discrepancies between the real anatomy and the final graph representation. We review these issues as a means to motivate our error-correcting algorithms.

The details of the data acquisition stage depend on the structure of interest and the imaging modality applied, yet a few considerations apply in nearly all cases. A series of 2D or 3D grayscale images are acquired with a fundamental resolution limit set by the imaging modality (Tsai et al., 2003; Weber et al., 2008). Technical realities such as detector noise, signal scatter, refractive aberrations, and acquisition speed can degrade the quality of the image. Regions of incomplete, weak, or extraneous contrast, due to poor labeling, intrinsic contrast gradations, or shadowing effects can adversely affect the perceived connectivity of the structure in the grayscale data.

The segmentation stage involves the conversion from grayscale data to a binary mask that accurately represents the morphometry and topology of the vascular structure. In particular, many methods exist to segment 3D vessels from raw data (Kirbas and Quek,

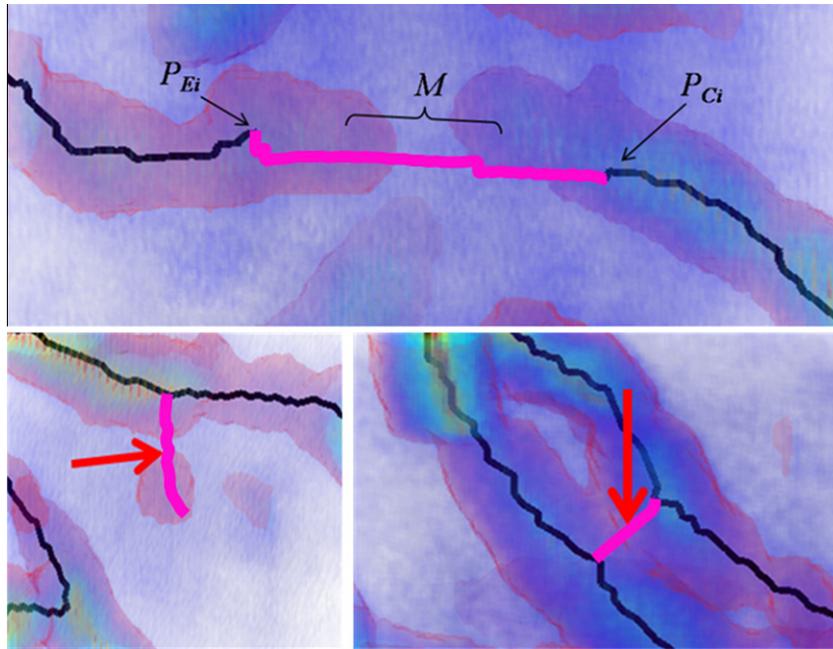
2004; Lesage et al., 2009), including multiscale eigenanalyses of local Hessian operators (Krissian et al., 1998; Frangi et al., 1998; Manniesing et al., 2006b), the use of filterbanks (Szymczak et al., 2005; Grady, 2010), tracking methods (Friman et al., 2010), vessel crawling (McIntosh and Hamarneh, 2006), level set-based curve evolution methods (Manniesing et al., 2006a) and segmentation learning approaches (Lu et al., 2009). Previous work on segmentation and vectorization of microvasculature from 3D optical image stacks (Cassot et al., 2009; Tsai et al., 2009b) demonstrate that high-fidelity segmentation can be achieved, particularly at high image contrast and signal-to-noise. Nonetheless, at regions of locally low image quality, segmentation errors persist (Fig. 2). Regions of partial dye occlusion, e.g., by a trapped blood cell, or weak staining can lead to breaks in the mask. Similarly, close-proximity vessels may result in erroneously joined masks, particularly in the limit of low signal-to-background, e.g., the presence of non-vascular structures in the data.

The vectorization stage entails the conversion from a binary mask to a graph or vector representation consisting of individual 1D strands and their interconnections (Risser et al., 2007a). A common first step in this process is the extraction of centerlines from the binary mask. Existing skeletonization methods can accomplish this for 2D datasets, but the ill-conditioned medial axis transform (MAT) that is fundamental to these techniques results in many small erroneous branches that develop as a result of either noise or local bulges along the structure itself. In 3D, MATs can also develop “medial surfaces” which are not centerlines at all. Curve evolution methods and morphological operators have been introduced to help mitigate these issues (Maddah et al., 2005; Leymarie et al., 2004; Bouix et al., 2005), and include recent work on vectorizing 3D microvascular networks (Risser et al., 2007b; Cassot et al., 2006; Maddah et al., 2005; Tyrrell et al., 2006). Still, discrimination between short outcroppings that require representative strands and bulges that result in erroneous hairs remains an unsolved issue.

Beyond the prior image analysis work on vessels, many biomedical imaging applications treat the analysis of filamentous structures and these may benefit from some of the methods described here (Gillette et al., 2011). For instance, the DIADEM challenge winners (<http://diademchallenge.org/>) developed the FARSIGHT toolkit for semi-automated tracing and vessel segmentation (Bjornsson et al., 2008). One interesting aspect of their approach (Bjornsson et al., 2008) is that automated analysis results are scored and manually corrected in rank order; similar use of confidence is envisioned for this work. The proof-editing of reconstructed neurons in the DIADEM challenge was the key bottleneck to produce accurate 3D reconstructions at scale (Peng et al., 2011). A recently developed 3D WYSIWYG interface to volumetric data for neuron tracing that allows editing the connectivity in 3D aims to address this bottleneck (Peng et al., 2011).

## 1.3. Published gap-connection methods

The problem of connecting gaps in 2D, or weak edge linking, is a rich and well-established field of study (Casadei and Mitter, 1996; Canny, 1986) and includes some recent studies on inferring missing edges in road network extraction (Sharon et al., 2000; Ren et al., 2007; Rothwell et al., 1994; Poullis et al., 2008). Similarly, edge deletion using machine learning to remove false edges from an overcomplete set has been demonstrated on the Berkeley Segmentation Benchmark (Kaufhold and Hoogs, 2004). Far fewer results have been collected for the analogous 3D problems (Krissian et al., 1998; Pock et al., 2005). One promising connection method grounded in the formalism of tensor voting uses the vectorized graph alone to infer gap connections (Risser et al., 2007a). More recently, the method has been shown to perform favorably to



**Fig. 2.** Filling gaps in the vasculature map. The top panel shows a gap formed in a portion of the vascular map. The missing section of vasculature, denoted  $M$  in this figure and colored magenta, is connected by a “bridging strand.” One goal of this work is to find all such gaps and compute an appropriate bridging strand,  $S_i$ , for each, that reflects the true topology of the biology, i.e., add an appropriate set of strands to the vascular graph. The bottom left panel shows a false positive “hair” in a portion of the vascular map. The hair is highlighted in magenta. The island to which the false positive hair leads, shown in light red, is a result of erroneous segmentation of a non-vascular structure in the imaged tissue. A second goal of the present work is to delete an appropriate set of “hairs” from the vascular graph. The bottom right panel shows a false positive spurious connecting strand in a vascular graph. The false positive strand is highlighted in magenta. These two strands are approximately orthogonal to the spurious connecting strand and do not physically connect in the specimen. A third goal of the present work is to delete an appropriate set of “spurious connections” from the vascular graph. All magenta strands of interest in this work have been retouched for contrast and clarity.

mathematical morphology and an Ising model for the same task (Risser et al., 2008). While the tensor voting method shows considerable promise, it relies only on the graph, and thus does not make use of the underlying grayscale data to inform the gap filling method and has also not yet been extended to exploit machine learning.

The methods in this paper learn to add or delete strand objects in vascular graphs, where these objects are results of other lower level image analysis algorithms. Similarly, related work in connectomics formerly aimed at optimizing voxel-level cluster similarity via the Rand index (Turaga et al., 2009) or a topology-preserving index (Jain et al., 2010a) have recently progressed to learning to cluster superpixels (Jain et al., 2011). Arguably, these superpixel approaches could also be applied to connecting gaps.

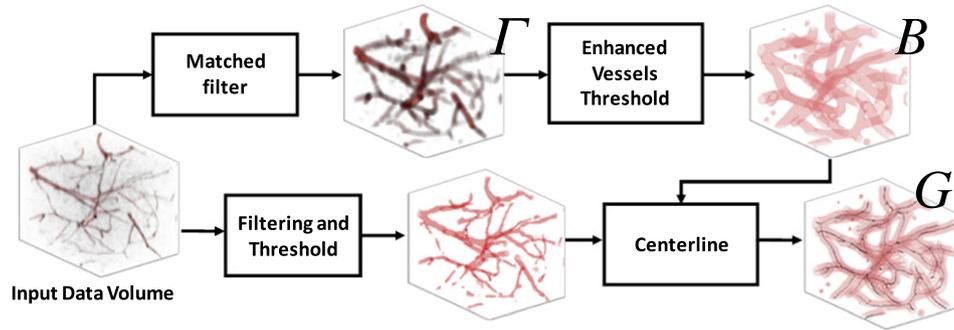
## 2. Methods

### 2.1. Microvascular datasets

Data analyzed in this paper were obtained using previously described protocols (Tsai et al., 2009b). In brief, mice are transcardially perfused with a low viscosity gelatin solution containing the fluorescent label, fluorescein-conjugated albumin. The solution solidifies by cooling and is fixed with paraformaldehyde to prevent dye diffusion. The brain is extracted and blocked into  $\text{mm}^3$ -scale tissue blocks which are then imaged with all-optical histology using two photon laser scanning microscopy (Tsai et al., 2003). This produces hundreds to thousands of overlapping 3D image stacks of fluorescein-stained microvasculature and surrounding tissue structures. These image stacks are nominally aligned based on approximate microscope stage coordinates and then aligned to voxel resolution by minimizing a quadratic registration energy in overlap regions across all blocks (Preibisch et al., 2009). This produces a large raw grayscale volume, typically  $1000 \times 1000 \times 1000$  voxels for our preparation.

A suite of image analysis algorithms (Tsai and Kaufhold, 2009a; Tsai et al., 2009b) is applied to selectively filter this block of raw, noisy vasculature data, segment it into a 3D binary mask, and extract a set of monofilament centerlines. Vectorization of the centerline results in a graph,  $G = (V, E)$ . Although not the main focus of the work in this paper, the main components of the image analysis algorithms applied prior to the learning in this paper include modules to enhance vessels, create volume-accurate vessel masks, create topologically accurate vessel masks and compute vessel centerlines (Fig. 3). These stages represent the degree to which data has been processed prior to the strand addition and deletion learning algorithms described here. The quality of the perfusion, fluorescein concentration, laser settings, sectioning, and registration stages can lead to variation in image quality between datasets. In this work, for instance, dataset A is significantly reduced in SNR compared to the others. In addition to these acquisition variables, the strand addition and deletion learning algorithm performances will depend on a number of image analysis parameters above, including the rod filter length, various filter sizes in the mask-generation stages, various adaptive thresholds, and centerlining parameters. The centerlining stage has perhaps the most direct influence on the threshold relaxation performance because it directly controls the number of degree 1 endpoints in the graph,  $G$ . Adjustable parameters in the centerlining stage affect the resultant graph by rejecting small fragments of the vascular network that would otherwise form spurious “Islands” of vasculature or “hairs”.

In the vascular graph,  $G = (V, E)$ , each vertex  $v_i$ , with  $\text{deg}(v_i) \neq 2$ , is either a bifurcation point or an endpoint, while each edge connects two vertices and is attributed with associated radii estimates,  $r_i \in \mathbb{R}$  (Table 1A). This graph representation, along with the intermediate masks and grayscale volumes serve as a starting point for the work described here. Four such vectorizations, one for each of four distinct samples A through D, are analyzed in this work. Nearly the entire vascular mask is correctly segmented and fully



**Fig. 3.** A block diagram of upstream image analysis modules. The input data volume is the output of the stitched blocks after registration, and the output is the vascular graph,  $G$ . The “Matched Filter” produces a continuous-valued enhanced volume,  $\Gamma$ , from the raw volume, where thin, rod-like structures are enhanced relative to background; this has the effect of closing very small gaps in the raw grayscale data. From  $\Gamma$ , an enhanced binary volume mask,  $B$ , is computed which is nominally a binary mask in the vicinity of vessel lumens which enjoys the improved continuity properties of  $\Gamma$  that result from the matched filtering, but is not suitable for accurate estimation of vessel radii. A binary mask is also computed by filtering and thresholding which more accurately captures cross-sectional measures useful for computing vessel radii, but lacks the topological continuity properties of  $B$ . Together, the two masks are used to compute a centerline mask that has the same topological properties of  $B$ , centered in the mask. The vectorized representation of this centerline mask is the vascular graph,  $G$ , discussed below.

**Table 1**  
Symbols.

A. Symbols used in learning	
$\Gamma$	Continuous-valued grayscale vascular volume (an input)
$B$	Binary segmented vascular mask computed from grayscale volume (an input)
$G = (V, E)$	Vascular graph of vertices, $V$ , and edges between all vertices, $E$ (a vectorization of $B$ )
$S_i$	$i$ th bridging strand found by threshold relaxation
$C_i$	$i$ th bridging strand confidence (z-score) found by threshold relaxation
$\Phi_{BS}$	Set of features of all bridging strands found in a dataset
$m$	Number of bridging strands in a dataset
$\Phi_{DCS}$	Set of features of all deletion candidate strands in a dataset
$m'$	Number of deletion candidate strands in a dataset
$n_{MC}$	Number of Monte Carlo rounds of bagging
$n_T$	Number of rounds of boosting
$n'$	Number of resampled examples for bagging
$H(k)$	Learned (boosted, bagged, validated) score for $k$ th hair
$B(k)$	Learned (boosted, bagged, validated) score for $k$ th bridging strand
$r(i)$	The radius of the $i$ th edge in $G$
$i$	An index over bridging strands
$k$	An index over bridging strands that share a common endpoint with a short strand
B. Symbols used in Threshold Relaxation	
$P_E$	All degree 1 endpoints in $G_V$
$P_{Ei}$	The $i$ th degree 1 endpoint in $P_E$
$B_B$	Binary mask in a bounding box about $P_{Ei}$
$\Gamma_B$	Grayscale volume in a bounding box about $P_{Ei}$
$G_B$	Vascular graph in a bounding box about $P_{Ei}$
$P_{Si}$	All vertices on the strand with $P_{Ei}$ except $P_{Ei}$
$R_B$	Binary rejection mask of all points closer to $P_{Si}$ than $P_{Ei}$
$\mu_{BG}$	Average value of $\Gamma_B$ in non-vessel voxels
$\sigma_{BG}$	Standard deviation of $\Gamma_B$ in non-vessel voxels
$V_C$	Candidate connection vertices in bounding box
$T_z$	Grayscale threshold on $\Gamma_B$ corresponding to z-score
$B_z$	Binary mask in bounding box corresponding to z-score
$D_C$	Volume of geodesic path lengths from $P_{Ei}$ to all points in $B_z$
$P_{Ci}$	Point in $V_C$ with the lowest path length distance to $P_{Ei}$
$C_i$	z-score of $i$ th bridging strand (i.e., $T_z$ producing $P_{Ci}$ )
$T_T$	Tightest threshold on $G$ that maintains a bridge between $P_{Ei}$ and $P_{Ci}$
$T_U, T_C$	Upper and lower thresholds in binary search for $T_T$

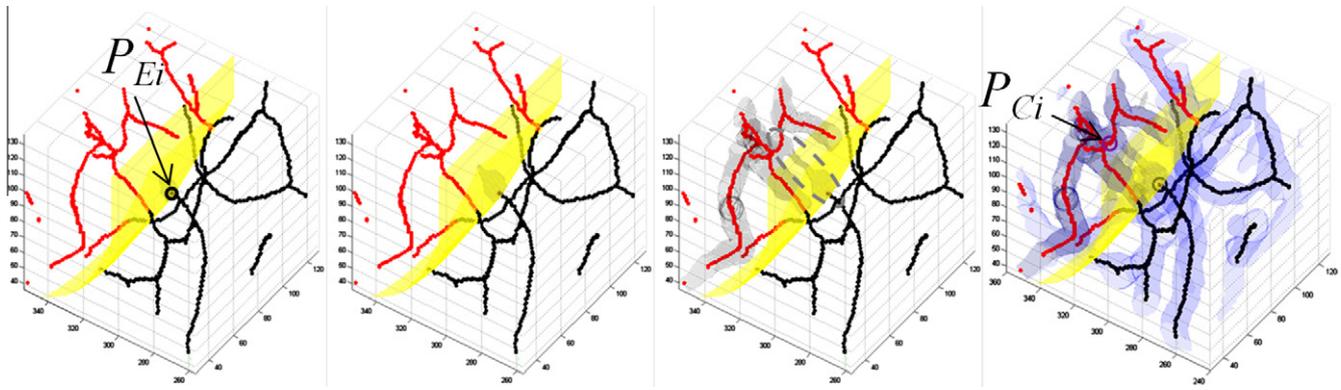
connected. Thus errors of any type are relatively infrequent. The number of gaps and spurious connections is small relative to the totality of strands in the vascular graph, typically numbering in the hundreds for the more than 10,000 strands in a  $1000 \times 1000 \times 1000 \mu\text{m}^3$  volume. Given the discussion above regarding the imperfect state of the art in segmentation and vectorization, and the sensitivity of topological metrics to graph structure, it is this population of graph defects which is the focus of the present work.

To describe the learning aspects of our method, the work presented here represents and describes operations on a number of

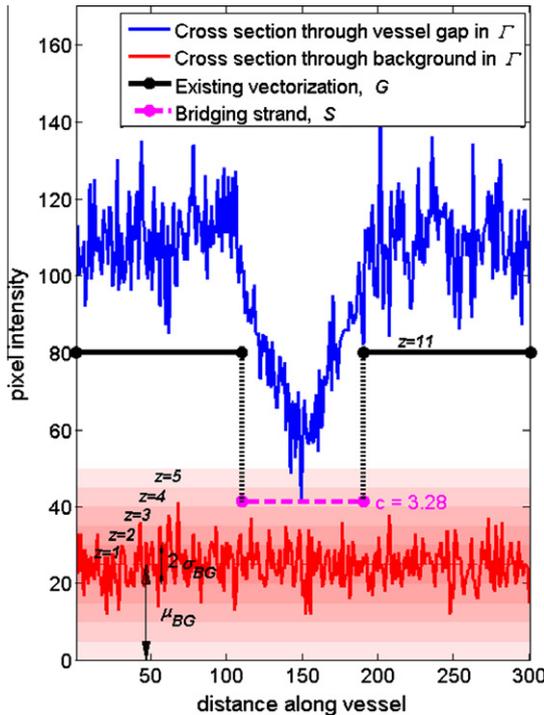
intermediate results of variable type and structure. These symbols are collected for reference in this section (Table 1A and B).

## 2.2. Bridging strand candidates

In this work, potential bridges to gaps in the vectorized strand network are found by a process of local threshold relaxation, but another algorithm that produces connections, e.g., tensor voting, could theoretically be used in its place. Tensor voting has been demonstrated to outperform an Ising model, a mathematical morphology model, and a standard phase field model of gap connection



**Fig. 4.** An illustration of the threshold relaxation process. At far left, the vectorization about the neighborhood of the endpoint,  $P_{Ei}$ , is shown with a black circle in the middle of the panel indicating the candidate endpoint. The red indicates connection candidates not ruled out by geometric constraints. The yellow dividing surface separates excluded (black) and candidate (red) centerline points from  $P_{Ei}$ . On the far left, the threshold,  $T_z$ , admits only a few points in the gap connecting mask,  $B_z$ . Moving right, the threshold is relaxed (lowered) and the mask contiguous with  $P_{Ei}$  admits more points until in the middle right panel, a connection is made to at least one candidate centerline point from  $P_{Ei}$ . Note the connection is made across the narrow segment of the black mask that did not originally contain a red centerline in the vectorization (indicated with a dashed gray ellipse). At the far right, the gap connecting mask (black) is superimposed on the original mask (blue). The black circle indicates  $P_{Ei}$  and the purple circle indicates the connection point,  $P_{Ci}$ .



**Fig. 6.** The confidence level for a 1D mock-up of the threshold relaxation process on synthetic data is the highest z-score compared to local background intensity statistics that would allow the gap to be bridged. A notional cross section along the axis of a vessel is shown in blue and a cross section from the nearby background is shown in red. The increasingly lighter red bars indicate z-scores above background spaced at increasing integer z values ( $\mu_{BG} = 25$  and  $\sigma_{BG} = 5$ ). In the mock-up, the endpoints of the vectorization found with a threshold that corresponds to  $z = 11$  is shown. Where the gap occurs, the threshold is reduced to  $z = 3.28$  before the notional 1D vectorization can be completed; thus the confidence for this bridging strand shown in dashed magenta is 3.28.

(Risser et al., 2008). An evaluation of precision-recall F-score measures for threshold relaxation was shown to outperform tensor voting for a range of parameterizations (Kaufhold et al., 2008).

At every internal endpoint in the network, a corresponding sub-volume of grayscale data is evaluated at progressively lower



**Fig. 5.** Algorithm accepts as input a binary mask of the microvasculature,  $B$ , a continuous-valued volume,  $\Gamma$ , and a graph,  $G$ , and produces a set of “bridging strands”,  $S_i$ , and their confidence levels,  $C_i$ .

thresholds until a connection is made to another strand, subject to geometric constraints. A confidence is assigned to each bridge based on the value of the final threshold relative to the background noise in the grayscale sub-volume (Figs. 4 and 6).

The threshold relaxation algorithm accepts as input a filter-enhanced grayscale image volume,  $\Gamma$ , the corresponding binary segmentation,  $B$ , and its vectorization,  $G$  (Table 1A). The output of the method is a set of bridging strands,  $S_i$ , and their respective confidence levels,  $C_i$  (Fig. 5).

2.2.1. Notional 1D example of threshold relaxation

The confidence level for any individual bridging strand is the highest z-score that would allow the gap to be bridged through a path between an endpoint in the vascular graph and another point in the vascular graph not excluded by geometric constraints (Fig. 6).

2.2.2. Threshold relaxation pseudocode

Gap connection proceeds in two stages (pseudocode in Fig. 7). First, candidate endpoints in the graph are found for each gap (Figs. 4 and 7A). Every gap is presumed to originate at an endpoint vertex,  $P_{Ei}$  (Table 1B). Second, a monofilament strand is found that bridges these points. In a local volume about the endpoint, a threshold,  $T_z$ , is applied to the grayscale volume,  $\Gamma$ , to produce a new binary mask,  $B_z$ . Geometric constraints are applied to avoid backtracking by excluding points that fall behind the endpoint vertex. The threshold is relaxed until the resulting binary mask produces at least one connection between the endpoint vertex and at least one other graph point,  $P_{Ci}$ , in the space not rejected by geometric constraints. If multiple points on the graph become joined at a particular  $T_z$ , then the point that minimizes the mask-constrained path length to the endpoint vertex is chosen to be the connection point,  $P_{Ci}$ .

(A) Function $[S, C] = \text{ThresholdRelaxation}(B, \Gamma, G)$		
1:	<b>Given</b> $B, \Gamma, G=(V, E)$ <b>Find</b> $S, C$	Bridging Strands, $S$ , and their confidences, $C$ , are computed from the inputs as in Fig 5.
2:	$P_E = \{\text{all interior degree 1 vertices in } V\}$	$P_E$ is the list of all interior endpoint vertices in $G$
3:	<b>for all</b> endpoints, $P_{Ei} \in P_E$ <b>do</b>	For each endpoint vertex, $P_{Ei}$
4:	<b>GetBoundingBox</b> $(P_{Ei}): B \rightarrow B_B, \Gamma \rightarrow \Gamma_B,$ $G \rightarrow G_B$	Restrict each input, $X \in \{B, \Gamma, G\}$ , to its local bounding box centered about $P_{Ei}$ . Subsequent operations are applied to the $X_B$ versions of each input.
5:	$P_{Si} = \{\text{Points on strand containing } P_{Ei} \setminus P_{Ei}\}$	$P_{Si}$ are all other points on the strand with $P_{Ei}$ except $P_{Ei}$
6:	$B_R = D_{\wedge}(B_B \setminus P_{Ei}) < D_{\wedge}(B_B \setminus P_{Si})$	The rejection mask, $B_R$ is false for points in $\Gamma_B$ closer to $P_{Si}$ than $P_{Ei}$
7:	$\mu_{BG} = \mu(\Gamma_B \mid B_B = 0)$	The intensity mean in $\Gamma_B$ in nonvessel voxels
8:	$\sigma_{BG} = \sigma(\Gamma_B \mid B_B = 0)$	The intensity sigma in $\Gamma_B$ in nonvessel voxels
9:	$V_C = \{V(G_B) \setminus P_{Ei}\}$	$V_C$ are all the candidate vertices in $G_B$ except for $P_{Ei}$
10:	<b>initialize</b> $z = (\max(\Gamma_B) - \mu_{BG} + \Delta) / \sigma_{BG}$	Start the $z$ threshold at the maximum intensity in $\Gamma_B$
11:	<b>while</b> $P_{Li} = \emptyset$ <b>do</b>	
12:	$T_z = \mu_{BG} + z \sigma_{BG}$	Compute an intensity threshold corresponding to $z$
13:	$L_z = (\Gamma_B > T_z) \wedge B_R$	A mask of all points in $\Gamma_B$ above $T_z$ and allowed by rejection mask (i.e. points in “front” of $P_{Ei}$ )
14:	$B_z = L_z$ contiguous with endpoint $P_{Ei}$	Find only the connected component of $L_z$ with $P_{Ei}$
15:	$P_{Li} = \{V_C \mid B_z(V_C) = 1\}$	$P_{Li}$ is a list of all candidate connection points in $B_z$
16:	<b>if</b> $P_{Li} \neq \emptyset; T_{Ci} = T_z; \text{return}$	If any candidate connection point is found, return the threshold
17:	<b>else</b> $T_{Ui} = T_z; \text{relax } z$	If no candidates are found in $B_z$ , lower $z$ and repeat
18:	<b>end while</b>	
19:	$D_C = D_P(B_z, P_{Ei})$	Geodesic distances to all candidate points restricted to $B_z$ (Appendix A)
20:	$P_{Ci} = \arg \min P_{Lij} D_C(P_{Lij})$	Choose the point with the shortest geodesic distance to $P_{Ei}$ as $P_{Ci}$
21:	$[S_i, C_i] = \text{GetBridgingStrand}(\Gamma_B, P_{Ei}, P_{Ci},$ $T_{Ci}, T_{Ui})$	Tighten threshold and find shortest path from $P_{Ei}$ to $P_{Ci}$
22:	<b>end for</b>	

(B) Function $[S_i, C_i] = \text{GetBridgingStrand}(\Gamma_B, P_{Ei}, P_{Ci}, T_{Ci}, T_{Ui})$		
1:	$T_T = \text{ChooseTightThreshold}(\Gamma_B, P_{Ei}, P_{Ci},$ $T_{Ci}, T_{Ui})$	Binary search finds a tight threshold, $T_T$ , connecting $P_{Ei}$ to $P_{Ci}$
2:	$B_T = \Gamma_B > T_T$	$B_T$ is a tight mask connecting $P_{Ei}$ to $P_{Ci}$
3:	$C_i = z(T_T) = \frac{T_T - \mu_{BG}}{\sigma_{BG}}$	The bridging strand confidence is the $z$ -score corresponding to $T_T$ .
4:	$D_G = D_P(B_T, P_{Ei}) + D_P(B_T, P_{Ci})$	$D_G$ is the sum of geodesic distances from $P_{Ei}$ and from $P_{Ci}$ , creating a valley of approximately constant sum distance at the points participating in the minimum distance path.
5:	$B_G = D_G > \min(D_G) + \Delta$	$B_G$ is a mask connecting $P_{Ei}$ to $P_{Ci}$ restricted to the minimum distance path plus a small constant
6:	$S_i = \text{Dijkstra}(B_G, P_{Ei}, P_{Ci})$	The bridging strand is the shortest path from $P_{Ei}$ to $P_{Ci}$ along $B_G$

**Fig. 7.** (A) Threshold relaxation pseudocode (see Table 2A): Interior vertices are those vertices further from the data bounds of  $\Gamma$  than a distance  $S_E$  ( $S_E$  is 45 voxels in this work).  $\Delta$  is a small positive constant. The vertices of the graph,  $G$ , are located at  $V$ . (B) Get bridging strand pseudocode:  $D_G$  is the continuous-valued “paired path length distance” in the gap between  $P_{Ei}$  and  $P_{Ci}$ . Note that  $B_G$  was derived from  $B_T$  (Fig. 8). The  $i$ th “bridging strand,”  $S_i$ , is computed via Dijkstra’s shortest path algorithm from  $P_{Ei}$  to  $P_{Ci}$  constrained to  $B_G$  (Fig. 8).

In the second stage, the strand that bridges the gap is found. Two steps are performed to refine the binary mask,  $B_z$ . First a binary search over thresholds is used to tighten the mask to include the fewest voxels while still maintaining a connection. Second, a “paired path length distance transform” is used to eliminate all points in the mask except those most likely to participate in the path. In brief, the shortest path connecting  $P_{Ei}$  and  $P_{Ci}$  in the resulting smaller mask,  $B_G$ , computed with Dijkstra’s algorithm, is the output strand,  $S_i$ .

The second stage of the threshold relaxation algorithm makes use of the endpoint, connection point, thresholds, and local volumes found in lines 1 to 20 of the pseudocode (Fig. 7A) and is encapsulated by the call to `GetBridgingStrand` in line 21 of the code. In line 1 of `GetBridgingStrand` (Fig. 7B), the tight threshold,  $T_T$ , is chosen via a binary search of thresholds between  $T_{Ci}$  and  $T_{Ui}$  such that  $B_T(P_{Ei}) = 1$ ,  $B_T(P_{Ci}) = 1$  and  $D_P(B_T, P_{Ei}, P_{Ci}) < \infty$ , i.e.,  $P_{Ei}$  and  $P_{Ci}$  are connected via  $B_T$  (Fig. 8).

The output of the threshold relaxation algorithm is the set of candidate bridging strands,  $S_i$ , and their corresponding confidence metrics,  $C_i$ . Each candidate bridging strand is a sequential list of coordinates of the monofilament curve which spans the gap. The confidence metric is the statistical  $z$ -score of the refined threshold used to generate the bridging strand. Note that threshold relaxation is the algorithm applied in this case, but theoretically any

algorithm that made connections on endpoints could be used (such as Tensor Voting (Risser et al., 2007a), methods based on Ising models (Risser et al., 2008), and morphological closing (Gonzalez and Woods, 2002), for example).

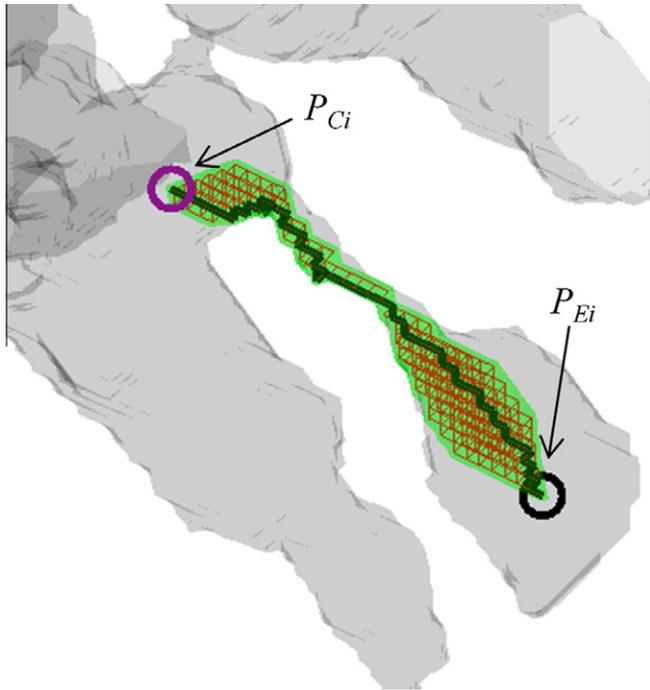
### 2.3. Deletion strand candidates

Candidates for strand deletion were chosen based solely on strand length. All strands that fell below a threshold length of  $15 \mu\text{m}$  were considered candidates for strand deletion. At this threshold, approximately 17% of all strands were flagged as deletion candidates.

### 2.4. Machine learning implementation

#### 2.4.1. Learning: Candidate strand ground truthing

A human operator is presented with a tool for ground truthing as a means to produce a training set for the machine learning process. The operator visually considers each candidate strand in its local 3D context, including volume renderings of the grayscale data, the binary mask, and the vectorized graph. The operator then labels each candidate strand as a true or false candidate for bridging or removal. The necessity of training sets in which each candidate is ground-truthed by a human operator motivates the



**Fig. 8.** Bridging strand generation. The tight mask,  $B_T$  (gray mask) that bridges the gap between  $P_{Ei}$  (black circle) and  $P_{Ci}$  (purple circle) is shown in gray. The candidate voxel lattice edges (thin red lines) in the gap mask,  $B_C$  (green mask), are searched for the minimum path using Dijkstra's shortest path algorithm. The minimum path solution from  $P_{Ei}$  to  $P_{Ci}$  constrained to the gap mask is shown in the thick black curve. Lattice edges available to the search are shown in red.

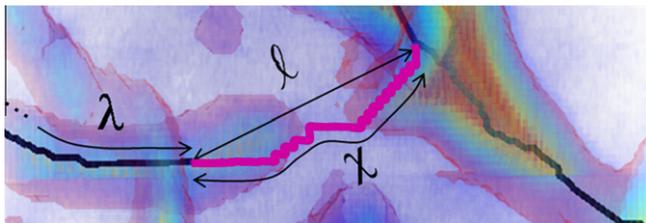
selection of only a sub-population of short strands as deletion candidates, rather than allowing all strands to serve as candidates; in theory all strands are deletion candidates.

#### 2.4.2. Learning: Bridging strand features

Basic threshold relaxation produces only a single confidence metric on each bridging strand based on the strand's intensity statistics (Fig. 9). For the purpose of machine learning, the threshold relaxation algorithm is extended to compute a number of additional features, including vertex count, curve length, etc. (Table 2B). These additional features are stored along with the original confidence metric in  $\Phi_{BS}$ . Note that each column of  $\Phi_{BS}$  corresponds to the features of an individual bridging strand.

#### 2.4.3. Learning: Deletion strand features

Just as an intensity-based confidence metric can be computed as the minimum z-score for bridging strands (Fig. 6), it can serve as a confidence metric for candidate deletion strands as well.



**Fig. 9.** Illustration of bridging strand features. A subset of the strand and strand neighborhood geometry metrics are shown for the magenta candidate strand. The Euclidean distance from the starting point of the bridging strand to the ending point of the bridging strand is  $l$ , the geodesic length of the bridging strand is  $\chi$ , the tortuosity is their ratio,  $\chi/l$ , and the length of the strand that the bridging strand originates from is  $\lambda$ . The operator that takes a strand as input and computes its length is  $L(S)$ , where  $S$  is the strand, e.g.,  $\chi = L(S)$  (Table 2B).

**Table 2A**

Feature sets. Functions and conventions in threshold relaxation pseudocode.

$\text{GetBoundingBox}(P_{Ei})$ :	Restricts the volume $X$ to only the volume in the bounding box, $X_B$ about $P_{Ei}$
$X \rightarrow X_B$	Binary mask of all 1s except at locations, $P$
$B_W \setminus P$	Euclidean distance from $P$ to all points in $B_W$
$D_r(B_W \setminus P)$	Mean of $E$ where $B$ is true
$\mu(E B)$	Standard deviation of $E$ where $B$ is true
$\sigma(E B)$	Pathlength distance from point $P$ to all points in mask $B$ , where all paths are restricted to lie within $B$
$D_r(B, P)$	The Dijkstra shortest path from $P_{Ei}$ to $P_{Ci}$ constrained to mask, $B$
$S = \text{Dijkstra}(B, P_{Ei}, P_{Ci})$	

**Table 2B**

Feature sets. The features,  $\Phi_{BS}$ , summarizing each bridging strand.

Strand intensity statistics	Confidence	$C$
Strand geometry	Vertex count	$v$
	Point to point length	$l$
	Curve length	$\chi$
	Tortuosity	$\tau = \chi/l$
Strand neighborhood	Profile distance slope	$P$
Incident strand geometry	Originating strand length	$\lambda$

Similarly, additional features beyond confidence are computed for each candidate deletion strand, including features based on intensity, geometry, and topology of either the deletion candidate strand itself, or the neighboring strands to which the candidate connects. The list of deletion strand features is stored in  $\Phi_{DCS}$  (Table 2C). Note that each column of  $\Phi_{DCS}$  corresponds to the features of an individual candidate deletion strand.

#### 2.4.4. Learning: Boosting and bagging strand features

AdaBoosted decision trees were used to score strands in this work. AdaBoost was run for  $n_T = 100$  (Table 1A) rounds for each training set using decision tree weak learners. In this analysis, the MATLAB® (R2007b) (MATLAB, 2009) Statistics Toolbox™ (Version 6.1, The Mathworks, Inc.) implementation of a decision tree was used. To prevent overfitting, the minimum example count per leaf node was set to 5, and the minimum splitting fraction was 10% of the total number of examples used to train each tree at each round of boosting.

The method of bootstrap aggregating, or bagging (Breiman, 1996), is used to improve the boosted decision tree model's stability, reduce its variance, and prevent overfitting of parameters. In bagging,  $n_{MC}$  (Table 1A) individual classification models are each trained (via boosting as described above) on a set of training data uniformly resampled with replacement. When classifying a new strand, the individual classifier results on that strand are averaged into a single score. The two key parameters in bagging are the number of iterations of bagging to combine,  $n_{MC}$  (Table 1A), and the number of examples,  $n'$  to uniformly sample from a training set. The number of labeled strand exemplars in our datasets are very few (only 300–400 strands per training sample) and can be unbalanced (for bridging strands found by threshold relaxation, typically only 20–50 bridging strands per  $\Phi_{BS}$  are incorrect, for example). For this reason, twice the number of elements are sampled to reduce the likelihood that resamplings are rejected for containing only positive exemplars. This procedure of bagging boosted decision trees is carried out for both bridging and deletion candidate strands and for all training/testing sets described in Section 2.4.6.

Our scoring convention is such that a high score for either bridging or candidate deletion strands indicate that strand should be kept, i.e., the strand is likely part of the vascular graph—and a low score indicates that strand should not be kept.

**Table 2C**  
Feature sets. The features,  $\Phi_{DCS}$ , summarizing each deletion candidate strand.

Strand intensity statistics	Mean $z$	$z_{\mu} = (1/n) \sum_k z(k)$
	Minimum $z$ (i.e., confidence)	$z_{\min} = \min(z)$
	Middle $z$ drop	$z_d = (z(1) + z(n))/2 - z(n/2)$
Strand topology	Endpoint	$\gamma = (\deg(S(1)) = 1) \vee (\deg(S(n)) = 1)$
	Number of incident strands	$n_c = \deg(S(1)) + \deg(S(n))$
Strand geometry	Vertex count	$v$
	Point to point length	$l$
	Curve length	$\chi$
	Tortuosity	$\tau = \chi/l$
Incident strand intensity statistics	Median neighbor strand intensity	$\rho_{\text{median}} = \text{median}(\rho)$
	Minimum neighbor strand intensity	$\rho_{\min} = \min(\rho)$
	Variance of neighbor strand intensity	$\rho_{\sigma} = (1/n_c) \sum_k (\rho(k) - \rho_{\mu})^2$
Incident strand geometry statistics	Maximum neighbor strand length	$A_{\max} = \max(A)$
	Mean neighbor strand length	$A_{\mu} = (1/n_c) \sum_k A(k)$
	Minimum neighbor strand length	$A_{\min} = \min(A)$
	Variance of neighbor strand length	$A_{\sigma} = (1/n_c) \sum_k (A(k) - A_{\mu})^2$

#### 2.4.5. Validation: Leave One Out Cross Validation (LOOCV) and cross training

When evaluating detection performance for a single dataset (e.g., computing the improvement in detection performance on dataset C based on the other labeled elements of dataset C), the method of LOOCV validation is used. LOOCV, a form of cross-fold validation (Kohavi, 1995), uses all the training data for performance estimation, prevents over-fitting, and can be used to derive confidence metrics on ROC and Precision Recall curves (Fawcett, 2004). Because the number of strands in  $\Phi_{BS}$  and  $\Phi_{DCS}$  is small, each bagged boosted classifier is trained on all but one of the strands and tested on the strand that was left out. In this way, strandwise bootstrap aggregated scores that do not train and test on the same data are generated for each bridging strand and deletion candidate strand within a dataset.

When evaluating detection performance across a number of different datasets, the performance estimates are valid without LOOCV because training and testing datasets are different by construction. With different datasets, the resampled training set in each iteration of bagging is used to train a classifier to evaluate all samples in the test dataset instead of generating a bagged LOOCV set of scores per strand as is necessary for single datasets. We cross train two ways in this work: (1) training on individual datasets other than the test set (this is the case when classifiers are learned on dataset C and used to classify dataset B, for example) and (2) training on all three datasets left out (this is the case when classifiers are learned from datasets A–C and used to classify dataset D, for example).

In summary, for each individual dataset,  $n_{MC} \times m \times n_T \sim 15,000,000$ , decision trees were trained in the computation of the bagged, boosted, and LOOCV bridging strand scores, where  $m \sim 300$  (Table 1A), and approximately 20,000,000 trees were trained in the computation of the deletion candidate strand scores, where  $m' \sim 400$  (Table 1A). At the conclusion of the learning process for both  $\Phi_{BS}$  and  $\Phi_{DCS}$ , pairs of scores that share a common endpoint are selected and indexed by  $k$ . Thus,  $k$  is an index over all short strand endpoints that also have a bridging strand. These are  $B(k)$  for the element of  $\Phi_{BS}$  corresponding to the  $k$ th endpoint and  $H(k)$  for the element of  $\Phi_{DCS}$  corresponding to the  $k$ th endpoint.

### 2.5. Consistency learning between added and deleted strands

#### 2.5.1. Graph consistency and conflict

Learning for bridging strand addition and deletion strand removal are treated thus far as separate decisions through separate

machine learning stages, but these decisions are not independent. Specifically, if a bridging strand connects a point on a deletion candidate strand, a conflict between the addition and deletion decisions can arise. The cartoon diagram (Fig. 10) illustrates both consistent decisions and decisions in conflict. A graph consistency and conflict approach is applied, aimed at fusing these two decisions into consistent decisions over all interacting deleted and added strands.

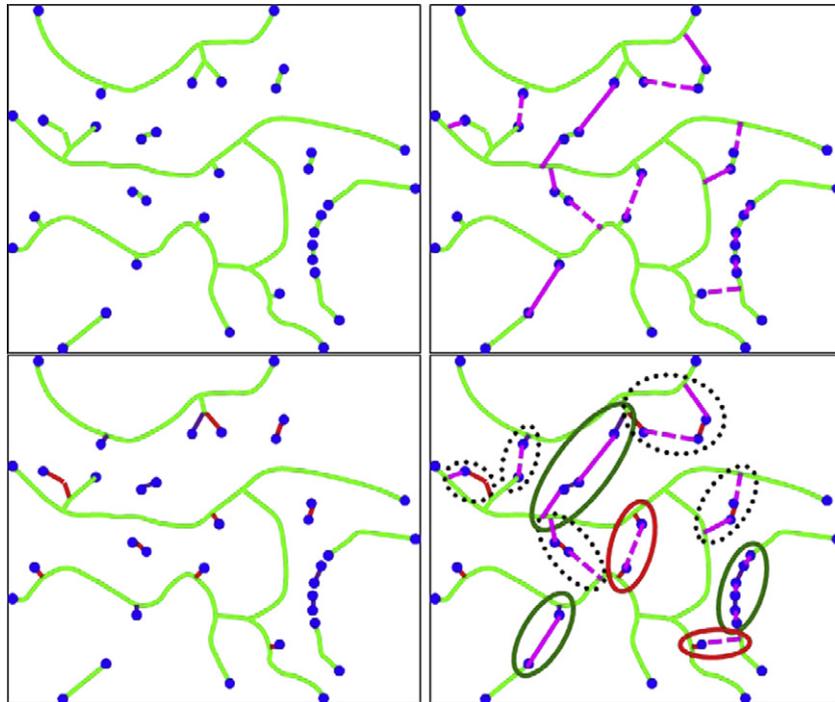
#### 2.5.2. Consistency: Learned strand consistency

Decision conflicts arise from endpoints on strands that are short enough to qualify as a deletion candidate. These short source strands will be associated with two learned scores: one for the bridging strand that emanates from it and one for the deletion candidate strand. The two scores,  $H(k)$  and  $B(k)$  for the  $k$ th endpoint, identify a point in the joint feature space that summarizes that endpoint's consistency. If both scores are high, then both the short source strand and the bridging strand are likely kept; if both scores are low, then they will both likely be eliminated. An example plot in the  $H$ - $B$  score space is shown (Fig. 11). The endpoints that are jointly labeled “keep” are shown with green “plus” signs, endpoints that are jointly labeled “remove” are shown with red<sup>1</sup> dots, and endpoints with conflicting labels are shown with black circles. A classifier is trained to separate the two types of consistent endpoints, and then it is applied to classify endpoints with conflicting labels.

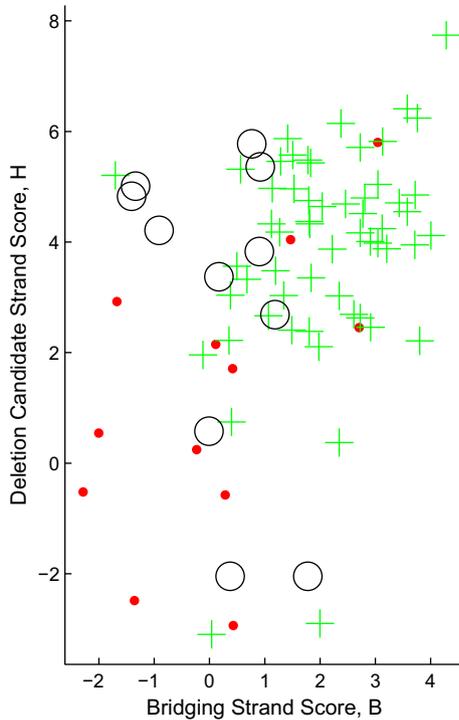
### 3. Results

A typical vascular graph in this work has 500,000 vertices, over 10,000 strands, and 40 vertices/strand. In a vascular graph of this size, there are typically 7500 junction points of degree 3 or higher and  $\sim 3000$  endpoints of degree 1. The majority of the endpoints occur where vessels leave the bounds of the 3D dataset, i.e., the data cube, while  $\sim 300$  occur internally. The vessels that create endpoints exiting the data cube are not considered for bridging or evaluated in performance measures. Thus, approximately 300 internal bridging strand candidates are generated per dataset. Additionally, based on a length threshold of 15 voxels, approximately 17% of all strands were considered for deletion.

<sup>1</sup> For interpretation of color in Figs. 1–4, 6, and 8–18 the reader is referred to the web version of this article.



**Fig. 10.** Cartoon of coupled decisions on bridging strand retention and spurious strand removal. The original vascular graph is illustrated in the top left panel with green strands and contains both gaps and spurious strands. Endpoints (degree 1 vertices) are blue dots. The results of learned threshold relaxation are shown in the top right panel. Solid magenta lines indicate retained bridging strands and dashed magenta lines indicate discarded bridging strands (based on ground truth). Note that all magenta bridging strands emanate from a degree 1 endpoint (blue dot), but that they may have a connection point that is not a degree one endpoint, a location on the green strands, for example. The bottom left panel indicates the decisions made on deletion candidate strands. The deletion candidate strands which are kept are shown in violet and those that are deleted are shown in red. The bottom right panel illustrates the coupling between bridging strand and deletion candidate strand decisions. Solid green ellipses illustrate consistent decisions to keep a putative subgraph. Solid red ellipses indicate consistent decisions to remove a putative subgraph. Dotted ellipses indicate that the decision to add a strand and delete a strand are in conflict. The aim of consistency learning is to provide a formalism to reason on these conflicting decisions where strand addition and strand deletion interact.

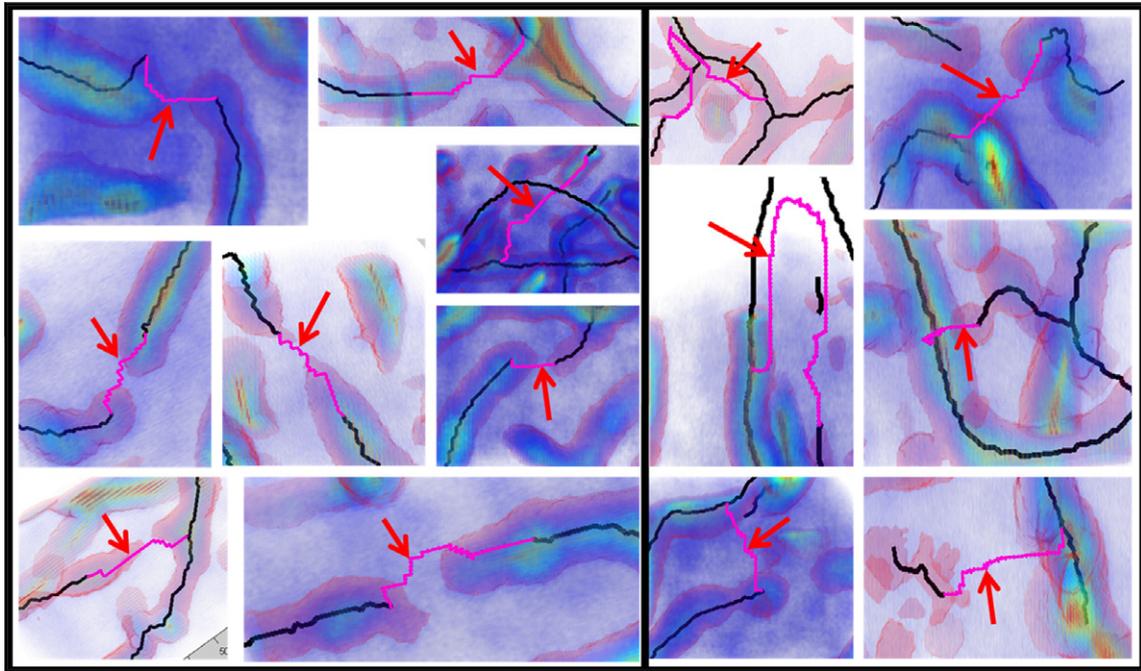


**Fig. 11.** A scatter plot of  $H$  and  $B$  scores for endpoints that share both a deletion candidate strand and a bridging strand. Endpoints with ground-truth that indicates both the bridging strand and the deletion candidate strand should be kept are labeled with green plus signs. Endpoints whose ground-truth indicates both the bridging strand and the deletion candidate strand should be eliminated are labeled with red dots. Strands with conflicting ground-truth labels are marked with black circles.

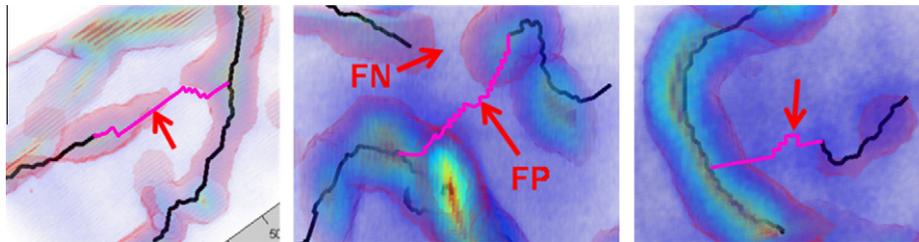
### 3.1. Learned threshold relaxation bridging strands

We first apply machine learning to the problem of bridging gaps in the vascular graph with the threshold relaxation method described above. Example bridging strands, computed by threshold relaxation are shown (Fig. 12). For the purpose of precision recall curve scoring, bridging strands are scored as a true positive, a false positive, or a false negative by a human operator (Fig. 13). A false negative constitutes an endpoint that the human operator believes should be connected to a strand other than the one bridged by the algorithm. These false negatives contribute both a false positive and false negative to the precision recall curve if the strand's score is such that the bridge is retained (middle panel of Fig. 13). If the bridge is not retained, only the false negative contributes to the precision recall curve. We compare the detection performance based on different features of the candidate bridging strand. A set of precision recall curves for all candidate bridging strands ( $m > 300$ ) in a dataset (left of Fig. 14). Curves are color coded according to the feature used to score the candidate strand. Note that the confidence computed by the original threshold relaxation algorithm does not dominate the detection performance, and thus does not constitute a dominant feature of the bridging strands.

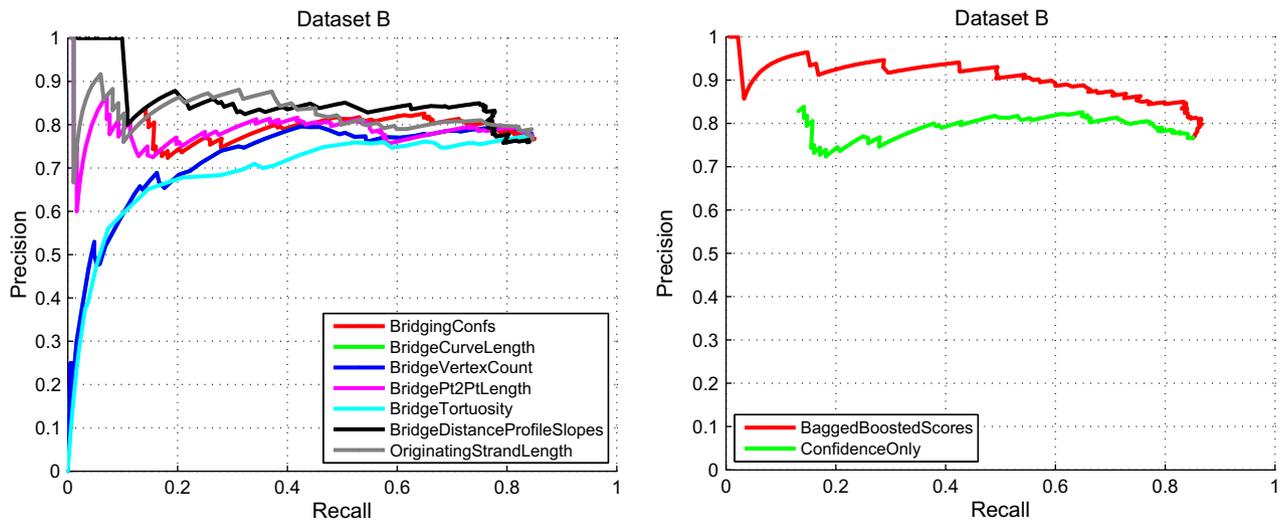
We next implemented machine learning, as described above, on the features produced by the threshold relaxation algorithm to produce bagged boosted scores for each bridging candidate. The original threshold relaxation confidence feature is compared to the detection performance learned classifier for an individual dataset (right of Fig. 14). For the example of Dataset B, the results indicate that the learned bridging strand scores dominate the detection performance compared to results without learning. The learned bridging strand scores also dominate the detection performance



**Fig. 12.** Correct and incorrect bridging strands. The left panel shows eight examples of correct bridging strands to add, as judged by a human operator. The right panel shows six examples of bridging strands that should not be added, as judged by a human operator. Note that the strands in the right panel either emanate from a strand that should be deleted or connect to the wrong connecting point.



**Fig. 13.** Bridging strand accounting. If a bridging strand is labeled as a false negative (FN) by a human operator, and its score is below the threshold, the bridging strand counts only as a FN. If its score is above the threshold, then the bridging strand counts as both a false positive (FP) and FN because the incorrectly connected bridging strand is a FP and the actual gap connection was still missed.



**Fig. 14.** Precision recall curves. Left panel shows the precision recall curves by feature. Note that the BridgingConfs feature, which is the confidence metric,  $C_i$ , computed by the original threshold relaxation algorithm, does not dominate the detection performance. Right panel shows the precision recall curves for the original threshold relaxation algorithm compared to the performance with machine learning. Note that the learning performance dominates detection performance of any individual feature on the left; the confidence precision recall curve is shown for reference.

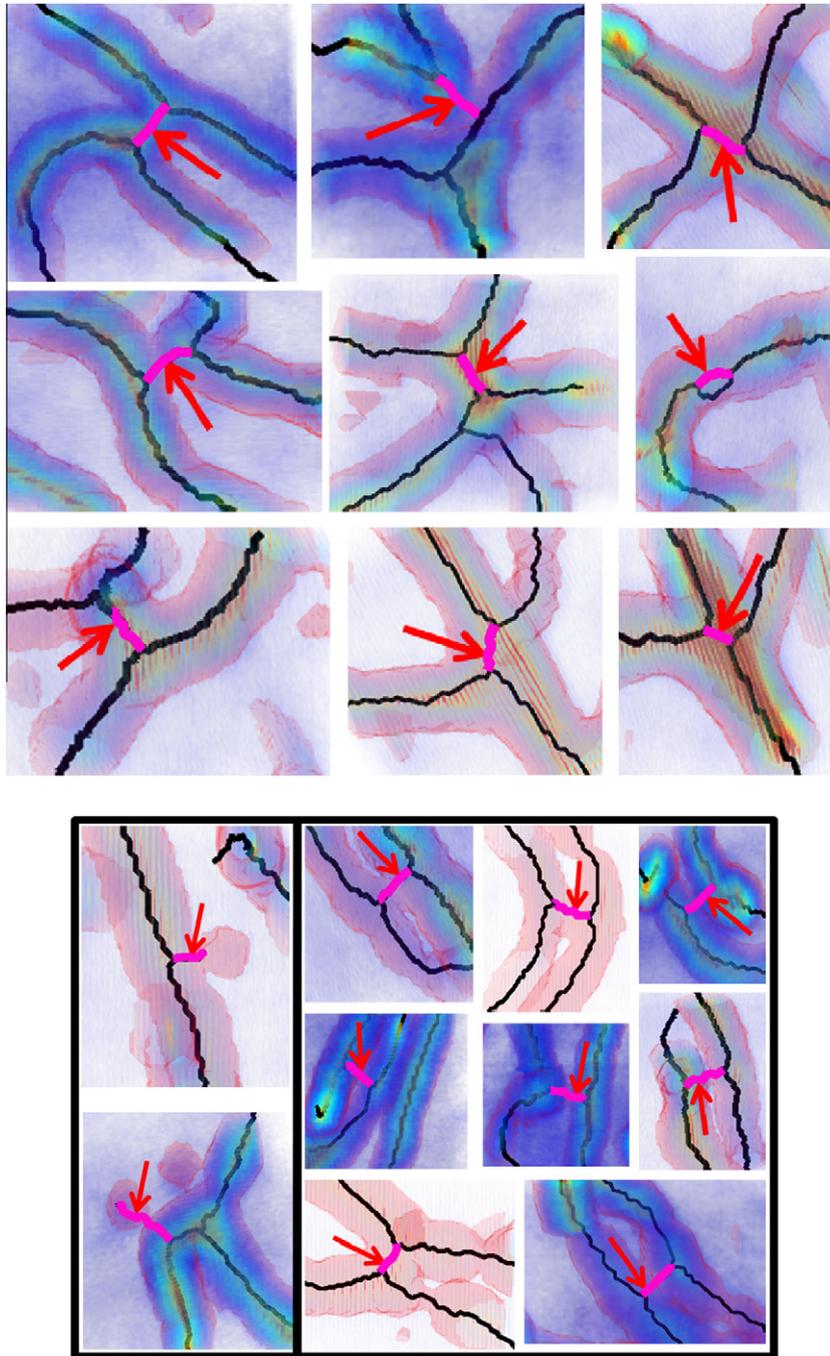
of all other individual features computed (comparison of the left and right panels of Fig. 14). This domination by the learned threshold relaxation method holds for all four datasets tested.

### 3.2. Learned deletion candidate strand decisions

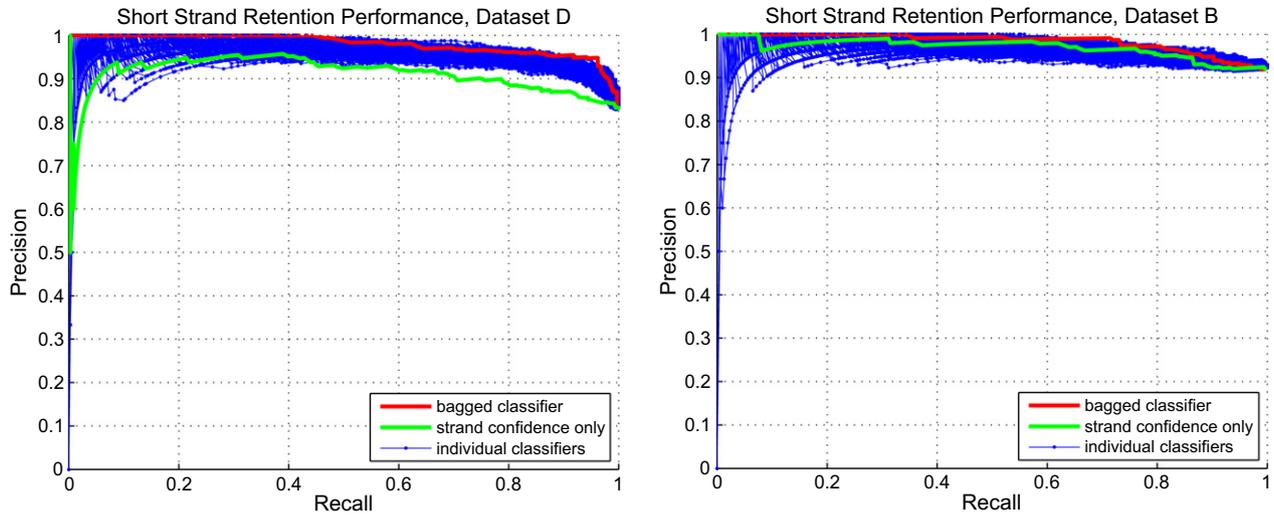
We also applied machine learning to the removal of spurious connections and hairs, defined as short strands that fell below a threshold length and marked as candidates for removal. Again, the machine learning algorithm was trained on a ground truth

subset of the data, in which removal candidates were judged by a human operator as strands that should be retained or deleted (Fig. 15). If a strand is correctly retained, it is a true positive. If it is incorrectly retained, it is a false positive. If a strand should be retained but is not, it is a false negative.

We consider the effect of bagging on the detection performance. Precision recall curves for the performance of learned candidate strand deletion performance demonstrate that bagging the boosted LOOCV score (Section 2) generally outperforms individual classifiers (Fig. 16). This indicates that LOOCV on a boosted,



**Fig. 15.** Deletion candidate strand labels. The top panel shows nine examples of deletion candidate strands that were judged by a human operator to be strands that should *not* be removed. The existing vascular graph is shown in black and the deletion candidate strand is highlighted with magenta dots and a red arrow. For the example, in the middle row and right column, the candidate strand could be more accurately collapsed with the parallel path in the very short vascular circuit and in fact is a vectorization error that results from an interior hole in the binary mask of vasculature, *B*; these collapsible strands are ignored in this paper. The bottom panels illustrate deletion candidates that a human operator judged should be removed. The two images in the left panel illustrate candidate “hairs” that were judged for deletion by a human operator. The eight images in the right panel illustrate candidate spurious connections that were judged for deletion by a human operator.



**Fig. 16.** Precision recall curves that demonstrate improvement in detection performance for deletion candidate strands with learning. Note that the performance with learning dominates the detection performance of the strand confidence ( $z_{\min}$ ) feature alone, but that performance improvement due to learning is dataset-dependent. Note also that the averaged classifier performance (in red) dominates nearly all individual classifier performances (in blue).

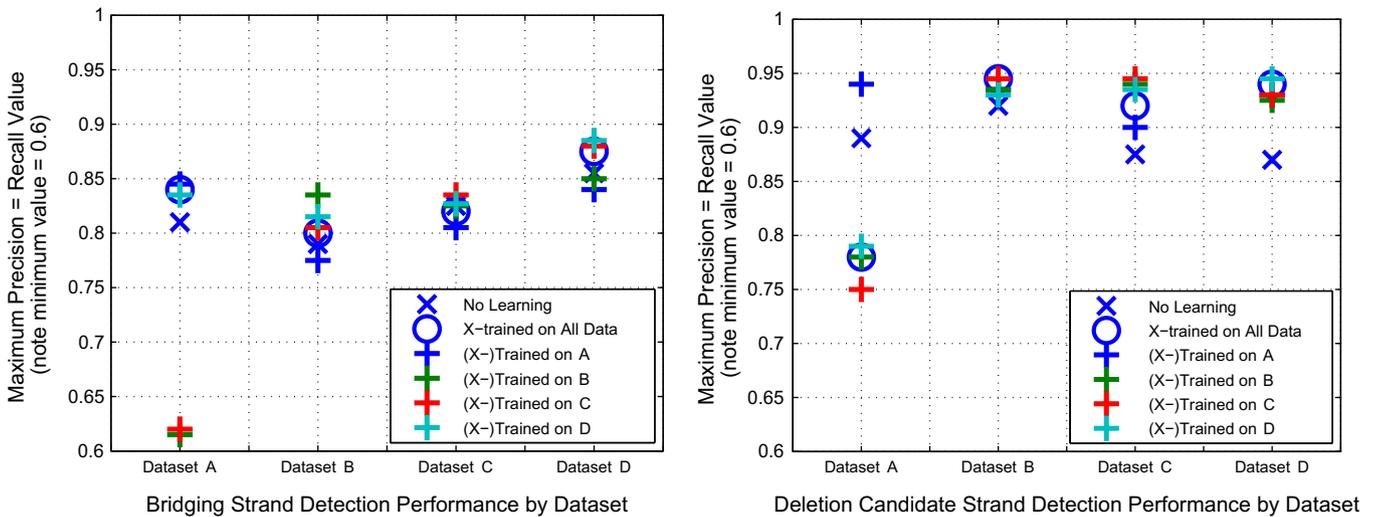
but not bagged, classifier may be sufficient to prevent overfitting, but that performance can be improved by bagging. The detection performance of the intensity-based deletion candidate feature,  $z_{\min}$ , compared to each deletion candidate's learned score illustrates the benefit of learning on multiple features compared to a more narrow feature-based approach (Fig. 16).

3.3. Empirical generalization error quantified

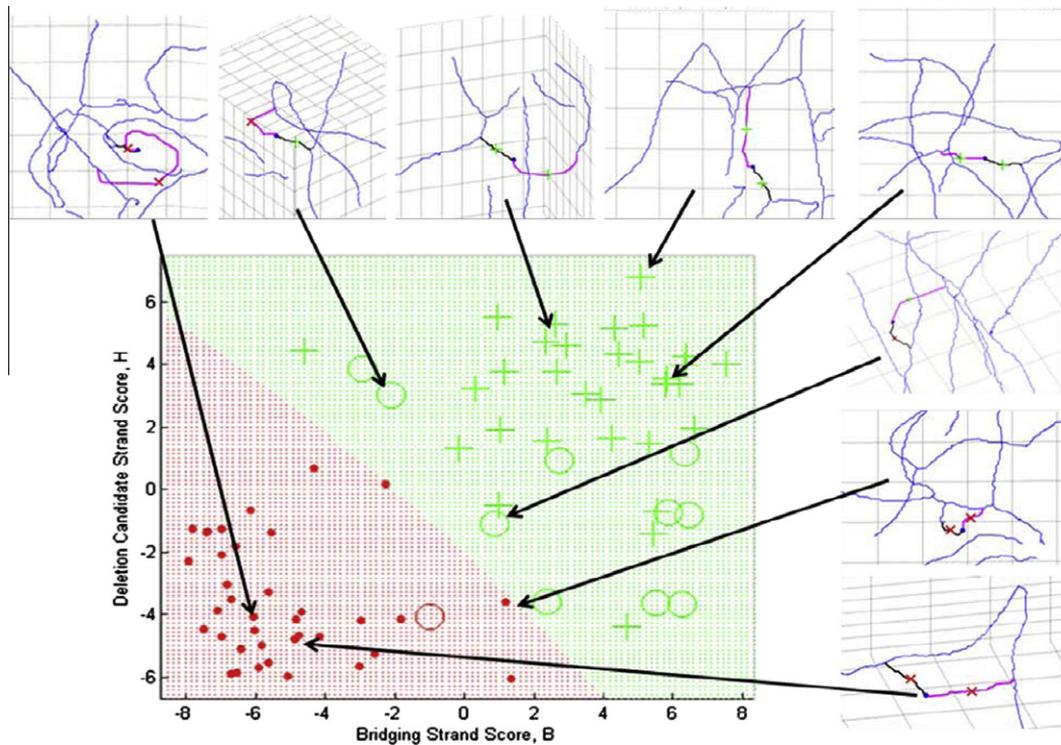
In our study of the vasculature, there is no known *a priori* reason to weigh false positives more or less than false negatives. Thus the intercept of a precision recall curve and the unity-slope line serves as an estimate of the overall performance of a learning algorithm on a set of data. A set of unity-slope intercepts of bridging strand detection performance curves are plotted for all datasets (left of Fig. 17). When the dataset in the column is the same as the training

dataset, LOOCV as described above is used for validation. The detection performance for deletion candidate strands is similarly presented (right of Fig. 17).

Note that in Section 2.4.1, only the shortest deletion candidate strands (<15  $\mu\text{m}$ ) were ground truthed. In theory, all strands in  $G$  could be false positives and this restricted sample could introduce sampling bias. To test the relative effect of this sample bias, a random sample of 500 deletion candidates between 15 and 40  $\mu\text{m}$  in length was also ground truthed. Only one of these 500 candidates was marked as a strand to delete; this error rate of approximately 0.2% is acceptable considering that all deletion candidate strand error rates are greater than 5% (right panel in Fig. 17). Note also that in the restricted sample of short strands that was ground truthed, more than 20% of deletion candidates <10  $\mu\text{m}$  in length and approximately 10% of deletion candidates between 10  $\mu\text{m}$  and 15  $\mu\text{m}$  in length were marked for deletion. This analysis by length



**Fig. 17.** Relative performance on different data sets. The panel on the left shows a set of precision–recall intercepts of bridging strand detection performance curves for each dataset examined. The column-wise performance estimates correspond to each dataset and include standard threshold relaxation, with an x per dataset, bridging strands learned from individual datasets, with colored plus signs, and bridging strands learned from all datasets except the dataset identified in the column, with circles. The panel on the right shows a set of precision–recall intercepts of deletion candidate strand detection performance curves for each dataset examined. The column-wise performance estimates correspond to each dataset and include performance of the deletion candidate strand confidence,  $z_{\min}$ , with an x per LOOCV dataset, deletion candidate strands learned from individual datasets, with colored plus signs, and deletion candidate strands learned from all datasets except the dataset identified in the column, with circles.



**Fig. 18.** Summary of performance of the consistency learning algorithm. The plot on the left shows consistency learning separates endpoints in the 2D space of scores for deletion candidate strands and bridging strands. A scatter plot of joint scores for endpoints with conflicting labels are shown with circles colored according to the results of consistency learning; green indicates a conflicting label was resolved as a pair of deletion candidate and bridging strands that should be retained to maintain consistency with other labeling decisions, while red circles indicate those endpoints associated with deletion candidate and bridging strands that should be removed to maintain consistency with other labeling decisions. Note that the red dots correspond to consistent decisions to remove strand pairs and green crosses correspond to consistent decisions to retain strand pairs—the red dots and green circles are used to train the classifier, from which the red/green boundary is inferred.

indicates that the ground truthing is focused on the candidate strands where learning is of value. All performance metrics are also computed only for the restricted set.

### 3.4. Consistency learning

Some endpoints produce simultaneous candidate strands for deletion and addition. As described in the Methods section, such strands are associated with two scores,  $H(k)$  and  $B(k)$  for deletion and gap bridging, respectively. A decision surface that separates the consistently labeled and scored endpoints is learned from all the endpoints with consistent ground-truth labels, using quadratic discriminant analysis (Fig. 18). Conflicting labels, indicated by open circles, are relabeled according to the learned consistency rule. In this way, learning is used to arbitrate among possible strand addition and deletion decisions to impose a consistency among all joint decisions made from two different learned functions. Note that only the atomic labels associated with addition or deletion were used in Consistency Learning. The labels in the joint space are inferred.

## 4. Discussion

### 4.1. Learned threshold relaxation overview

The focus of this paper is on methodologies to correct erroneous gaps and spurious strands in a vascular graph (Figs. 1 and 2). In particular, machine learning is applied to a set of features derived from both intensity and topological characteristics of the dataset. First, a method of threshold relaxation is used to detect gaps and provide candidate bridging strands to add to the graph (Kaufhold

et al., 2008) (Figs. 2 and 4–8). The threshold relaxation method, itself, enjoys a number of desirable characteristics: (1) it exploits graph topology to locate defects; (2) it exploits underlying intensity data to guide connections; (3) it prevents backtracking, and (4) it can connect large gaps. Next, machine learning is applied to features extracted from the candidate strands (Figs. 9–17). This learning framework introduces several additional advantages, including (1) the incorporation of multiple features based on both intensity and topology; (2) the potential to incorporate more sophisticated search strategies, e.g., tensor voting (Risser et al., 2007a, 2008) as additional learning features; (3) the minimization of the parameter sensitivity in graph correction decisions; and (4) when used in conjunction with deletion candidate strands, the ability to balance conflicts between decisions to jointly add or remove strands.

### 4.2. The intensity-based confidence metric

Every bridging strand discovered by the original threshold relaxation algorithm was assigned a confidence metric (Kaufhold et al., 2008) (Figs. 6, 7 and 14 and Tables 1A and 2B). The bridging strand confidence for each strand is a measure of the “weakest link” in the intensity evidence between two points. Specifically, it is the path through the intensity evidence whose minimum z-score is highest (Fig. 6). As such, it may fail to connect breaks that are not supported by grayscale data, e.g., regions in which the staining in a vessel falls to background levels for stretches that exceed the width of applied filters. This situation is ameliorated by the incorporation of other features, including geometric and topologically based features, into the decision process through machine learning.

#### 4.3. Learned generalized confidence metrics of threshold relaxation

The deterministic bridging strand confidence alone reaches maximal equal precision/recall rates on the lowest signal-to-noise ratio datasets of 78% to 86% (Fig. 17). However, we show that a number of other metrics can improve prediction performance, including strand length, tortuosity, and distance profile to closest vessel, among others. It is worth noting the lists of bridging and deletion strand features are not exhaustive (Tables 2B and C). These additional features are meant to quantify simple strand characteristics beyond the intensity-based confidence metric that may be used to decide whether each strand should be kept or removed. Note that bridging strand and deletion strand features need not be identical or even similar. The precision recall curve improves significantly with the use of validated machine learning scores that were generated by LOOCV scores on bagged and boosted decision tree classifiers using these additional features. The generalized confidence metrics computed using machine learning methods improve the best equal precision/recall values for these data sets by about 5% (left of Fig. 17).

#### 4.4. Spurious strands deletion reduces false bridging with threshold relaxation

Spurious endpoints, or vectorization “hairs,” can lead to erroneous connections in the threshold relaxation procedure (Kaufhold et al., 2008). The additional algorithmic step of selecting candidate strands for deletion addresses this limitation (Figs. 2, 13 and 15). The decision to remove a candidate is learned from a set of topology- and intensity-based features of the candidate strand and of neighboring strands. As discussed above, only short strands were considered deletion candidates in this work, but this strand validation concept can be extended to include all strands, thus providing a probabilistic framework for inference on the vascular graph.

#### 4.5. Consistency learning

The implementation of a strand deletion process along with a strand addition, i.e., gap bridging, process introduces the potential for conflicting decisions. Specifically, a decision to remove an endpoint-terminated hair may be inconsistent with a decision to add a bridging strand from that endpoint. While the deletion candidate strands could, in theory, be removed first, and bridging strands estimated on the remaining vascular graph, or vice versa, to avoid cascaded errors, we instead choose to coordinate the addition and deletion of strands according to joint statistics on both because both algorithms produce uncertain labels on strands. In its current implementation, consistency learning makes use of two features, the bridging strand score  $B(k)$  and the deletion candidate strand score  $H(k)$ , to generate a decision surface (Figs. 11 and 18). Extension of the decision consistency concept to a larger feature space is straight-forward and can be accomplished within the same learning framework used to generate the individual scores.

#### 4.6. Upstream considerations

In some datasets, e.g., dataset B, the application of learning to the strand deletion decision added relatively little value in detection performance compared to the strand confidence feature alone. The comparatively larger improvement in another data set, e.g., dataset D, illustrates the sensitivity of learning performance to upstream considerations. These considerations include tissue preparation, image acquisition, and a number of image analysis parameters such as filter sizes, adaptive thresholds, and centerlining parameters. Centerlining parameters most directly affect the number of degree 1 endpoints in the graph by controlling rejection

of short strands that are either unconnected, i.e., “islands”, or singly-connected, i.e., vectorization “hairs.” Nonetheless, any segmentation and vectorization algorithm may serve as upstream sources for the graph correction algorithms described in this paper.

#### 4.7. Small loops

Note that small vascular loops were ignored in this work for two reasons; they are infrequent and inconsequential to the flow modeling work which motivates the learning algorithms proposed. In the context of our learning algorithms and the size of our datasets, the frequency of small loops introduced by threshold relaxation before learning is very low. For instance, across our four datasets, the number of small loops introduced by threshold relaxation can be bounded above at 3% because that is the total number of bridges connected by threshold relaxation; i.e., there are approximately 300 bridging strands added to approximately 10,000 strands total per dataset. Further, on a sample dataset, only approximately 5% of the bridging strands introduced small loops. On dataset B, 13 of 326 bridging strands created spurious loops before learning, for example. In sum, approximately 0.15% additional spurious loops will likely be introduced into a dataset due to learned threshold relaxation. We also do not attempt to detect and/or delete loops or parts of loops. From the flow modeling perspective that motivates this work, small loops only introduce an infrequent additional parallel path in flow modeling at some locations, and are very minor effects when compared to the vessel lengths they are incident to. The loops introduced by threshold relaxation are unlikely to affect any statistical measure on datasets this size, and further, very small erroneous loops are typically simple to detect and reevaluate when they do arise.

#### 4.8. Generalization error

Evaluating LOOCV detection performance across datasets A to D, learned threshold relaxation error rates are reduced between by 5% and 21%, with a mean of 16%, compared to threshold relaxation without learning. Similarly, deletion candidate strand detection performance error rates are reduced between 18% and 57%, with a mean of 44%. This demonstrates the potential for learning to improve detection performance substantially for both tasks. However, bridging and deletion candidate strand population feature statistics are variable, and training and testing on different datasets (cross-training) reduces the observed performance gain from learning. Cross-trained detection performance (Fig. 17) for bridging strands show some expected degradation in the error rate reduction to between –3% and 15%, with a mean of 7%, as compared to LOOCV detection performance. The negative error rate indicates that the learned result was poorer than the result without learning. It is important to note that dataset A differs substantially in its signal-to-noise ratio from datasets B–D, and was included to quantify learned performance in a significantly altered population of features. Surprisingly, despite its differences, bridging strand detection for dataset A improved the most on cross-training compared to no learning. For deletion candidate strands, datasets B–D benefited from learning when trained on a different dataset, reducing error rates 31% to 53% on those datasets, with a mean of 40%, while dataset A’s performance was significantly worse than performance without learning. The implication is that in dataset A, many valid deletion candidates were short, and thus fit the criteria for selection without learning; but the topology- or intensity-based features of these hairs differed significantly from those in other datasets. Thus learning on other datasets served to confound decisions on dataset A.

The empirical analysis of generalization error suggests that different datasets have different feature/label distributions. The degradation in performance when training and testing on different

datasets compared to validated learning performance on the same dataset points to at least two strategies to reduce generalization error. First, upstream improvements in data quality as well as improved algorithm parameter stability can both be expected to make datasets more uniform, and consequently, benefit more from learning which vascular graph strands to delete and/or add. However, even without improving upstream data quality or parameter stability, learned results may be improved by better matching between individual dataset feature/label population statistics. Cross-trained results by dataset indicate that some datasets appear to match other datasets with variable fidelity – using only the best matched data for training may improve performance. Similarly, selective sampling methods can also be applied to the set of all training datasets to tune a sampled dataset to more closely reflect the population of features observed in the dataset targeted for testing. Finally, in the case that dataset statistics differ not only by feature distribution, but by ground truth label, active learning methods can be used, in conjunction with sampling methods described here, to minimize the number of additional labels in the test set required to improve performance with learning some incremental amount.

## 5. Conclusions

This work applies machine learning to correct connectivity defects in the graph representation of a 3D vascular network. Threshold relaxation (Kaufhold et al., 2008) is used to detect gaps and generate candidate bridging strands, while strand length is used to select candidate strands for deletion. Machine learning is then applied to a number of intensity- and topology-based features to generate deletion strand and bridging strand scores. A decision surface is trained to resolve conflicting scores, yielding a consistent decision for every candidate defect. Machine learning generally showed improved performance against unlearned single feature discrimination. Error analysis based on leave one out cross validation for same-set training yielded 16% and 44% mean improvements for bridging and deletion respectively, compared to no training, and achieved precision recall values as high as 88% and 94%.

As described above, each dataset is different, but a  $1000 \times 1000 \times 1000 \mu\text{m}^3$  dataset typically contains more than 10,000 strands. In this graph, there will be approximately 300 internal endpoints, of which approximately 280 should be connected as judged by a human and approximately 20 whose strand should be eliminated. If learning on bridging strands yields  $P = R = 0.85$  (Fig. 17), there will be approximately 222 true positives found by learning, with about 39 false positives and 39 false negatives (Fig. 13). In the same dataset, approximately 1000 short deletion candidate strands will contain approximately 150 spurious strands, and assuming learning yields  $P = R = 0.92$  (Fig. 17), approximately 36 of these spurious strands will remain and 36 strands will be incorrectly deleted. Of the strands not interrogated for deletion due to selection bias, there may be an additional 20 spurious strands based on a 0.2% spurious strand rate observed for strands longer than deletion candidates. Thus, the learning methods described in this paper reduce 470 total errors to 170 total errors, more than halving the error rate, and therefore, human time spent manually correcting datasets, which is argued to be the bottleneck for similar efforts (Peng et al., 2011). The additional benefit of consistency learning is expected to further reduce this error rate.

For both of the learning problems studied, we discovered a number of informative strand features that lead to improved decision-making performance regarding which strands to add and which strands to delete from a real vascular graph. Both strand addition accuracy and strand deletion accuracy were demonstrated to improve using a limited set of features. A richer feature set that

includes more information on the local topological and geometric context, e.g., local or regional grammar on strands, can be expected to incrementally improve performance of both algorithms. Finally, we formalized a tradeoff measure of consistency between adding new candidate strands and deleting candidate strands.

Algorithmic elements presented here provide intermediate vascular graph correction tools which are part of a larger system to reason more globally over the intermediate labels and scores computed from these modules. For instance, the consistency learning method provides a natural interface to active learning (Freund et al., 1997b; Dasgupta et al., 2009). By finding the endpoints that exhibit the most conflict across strand addition and deletion scores, human operator effort can more efficiently be directed to labeling the elements of the vascular graph that lie on or nearest the decision margin of the consistency learning problem. The learning algorithms also provide a formalism to rank and present confidence-rated examples to a human operator. Such a formalism may be of particular value in managing the variable SNR within and across these large datasets.

## Acknowledgments

We thank Mahnaz Maddah for making available her 3D Chamfer<sub>3,4,5</sub> path length distance transform. We thank the anonymous reviewers for their careful read and insightful comments which substantially improved the quality and overall readability of this manuscript. We thank Uygur Sumbul and Sebastian Seung for their helpful discussions on using convolutional networks for topology preservation in dendritic trees. This work was supported by grants from the National Institutes of Health (EB003832, MH085499, and OD006831).

## Appendix A. Euclidean and path length distance transforms

The standard Euclidean distance transform,  $D_r(B)$ , yields the distance from every bright pixel to its closest dark pixel. Note that the distance transform is 0 everywhere outside the mask. 3D Euclidean distance transforms are used in this work. The path length distance transform,  $D_p(B, s)$ , returns the path length distance,  $D_p$ , from a chosen starting point,  $s$ , inside the mask. The path length distance is defined as the geodesic, i.e., shortest path, from one point in the mask to another point in the mask constrained such that all intervening edges are also in the mask. By definition, the path length distance between points on the mask and points outside the mask is  $\infty$ , i.e., they are not connected. Many methods can be used to compute the path length distance transform, including fast marching methods and Chamfer methods. In this work, we use the Chamfer<sub>3,4,5</sub> method for path length computation as defined in (Maddah et al., 2005) (and abstracted from Mahnaz Maddah's work).

## Appendix B. Ground truthing

The 3D extent of each candidate strand sets its 3D bounding box for visualization – a margin of  $15 \mu\text{m}$  about each bridging strand is the default setting for volume and surface renderings. Volume renderings were produced using (Conti, 2011). For each bridging strand, the user is presented with a 3D volume rendering of the data,  $\Gamma_B$ , overlaid with a rendering of the mask,  $B_B$ . The strands in the graph,  $G_B$ , within some distance margin of the bridging strand, are also plotted in the same coordinate system. This default behavior can be adjusted with a control panel that can increase or decrease the size of the visualized bounding box. The volume rendering can be toggled off, set to render in 2D or 3D, or made more or less opaque. The surface rendering can also be toggled off and can be adjusted in transparency. By adjusting a proximity thresh-

old, the local vascular network topology representation of centerlines can be expanded or made to include only those strands touching the bridging strand. The nearby centerline plots can also be toggled off.

The visualization of each candidate strand in its 3D local vascular network context can be rotated, zoomed, and translated. Using these features, the user can rotate the bridging strand into a view angle and zoom level that most clearly illustrates whether it is or is not a valid candidate strand. A 2D snapshot of each candidate strand is retained for later auditing. Audit images streamline the process of resolving differences between different ground-truth labels assigned by different users for various candidate strands. At the user's discretion, multiple audit images can be collected. The labeling of bridging strands and deletion candidate strands differ slightly, as described below.

### B.1. Labeling bridging strands

After viewing each bridging strand in its local 3D context, the human operator labels the strand as either a correctly connected gap, i.e., a true positive, a "hair", i.e., a false positive, or an endpoint that should be connected to the vascular graph, but threshold relaxation found the wrong connection point so that it is both a false positive and a false negative. The human operator can also label a bridging strand as ambiguous, in which case its label is not used in learning.

### B.2. Labeling deletion candidate strands

Deletion candidate strands are similarly labeled by a human operator. After viewing each deletion candidate in its local 3D context, the human operator labels the strand for either retention or removal, by selecting whether the strand and/or its neighbors should be kept, or the strand or part of it should be deleted. The human operator can also mark the strand as "not sure," in which case its label is not used in learning.

## Appendix C. Boosting

The interested reader can find more theoretical and implementation details of Adaboost in (Freund and Schapire, 1997a). For the  $i$ th bridging strand, AdaBoost accepts as input a feature vector,  $x_i$ , and a label which indicates whether or not ( $y = +1/y = -1$ ) the example is labeled as an accurate bridging strand by the human observer. AdaBoost produces a score,  $H(x_i)$  from this input. In this analysis, AdaBoost was run for  $n_T = 100$  rounds for each training set using a decision tree weak learner.

Decision trees are well known classification tools which repeatedly partition a dataset by thresholding multiple features on successively smaller subsets of the feature set. Caruana et al. (Caruana and Niculescu-Mizil, 2006) demonstrated that boosted decision trees generate highly robust classifiers across multiple types of datasets and performance metrics, providing broad evidence that absent any other information or intuition on choosing a learning algorithm for a specific problem, boosted decision trees produce high-quality results more frequently than competing algorithms, including random forests, logistic regression, Naïve Bayes,  $k$ -nearest neighbors, standard decision trees, bagged decision trees, neural nets, support vector machines, or other statistical methods. Specifically, Caruana (Caruana and Niculescu-Mizil, 2006) shows that boosted decision trees consistently have the best area under the receiver operator curve, which is the closest proxy to the precision recall detection metrics we aim to optimize in this analysis. This motivates the use of boosted decision trees in our work.

Decision trees are known to suffer from overfitting when grown to full size on a large, noisy dataset (Bramer, 2007). In the case of fully grown trees, each leaf node will 100% correctly classify some of the training examples, leading to a 100% detection performance on the train data, but may suffer terrible performance on a novel test dataset. Many decision tree fitting techniques and algorithms have been developed to mitigate these overfitting problems, e.g., implementations include Quinlan's C4.5 (Quinlan, 1993) and Breiman et al.'s CART (Breiman et al., 1984), and include techniques to halt growing the tree after some complexity is reached or to grow a tree to full size and then prune back decisions that are likely due to noise alone. In this analysis, the MATLAB® (R2007b) (MATLAB, 2009) Statistics Toolbox (Version 6.1) implementation of a decision tree was used. To prevent overfitting, the minimum example count per leaf node was set to 5, and the minimum splitting fraction was 10% of the total number of examples used to train each tree at each round of boosting.

Note that other learning methods were also benchmarked using 500 iterations of bagging as was done for boosted decision trees. Support Vector Machines, linear discriminants, boosted stumps and boosted linear discriminants were evaluated and none outperformed boosted decision trees for the same number of bags for either bridging strand or deletion candidate strand learning.

## References

- Bjornsson, C., Lin, G., Al-Kofahi, Y., Narayanaswamy, A., Smith, K., Shain, W., Roysam, B., 2008. Associative image analysis: a method for automated quantification of 3D multi-parameter images of brain tissue. *Journal of Neuroscience Methods* 170, 165–178.
- Blinder, P., Shih, A.Y., Rafie, C.A., Kleinfeld, D., 2010. Topological basis for the robust distribution of blood to rodent neocortex. *Proceedings of the National Academy of Sciences USA* 107, 12670–12675.
- Bouix, S., Siddiqi, K., Tannenbaum, A., 2005. Flux driven automatic centerline extraction. *Medical Image Analysis* 9 (3), 209–221.
- Bramer, M., 2007. *Principles of Data Mining*. Springer, London, pp. 119–134.
- Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J., 1984. *Classification and Regression Trees*. Wadsworth, California.
- Breiman, L., 1996. Bagging predictors. *Machine Learning* 24 (2), 123–140.
- Canny, J., 1986. A computational approach to edge detection. *PAMI* 8 (6), 679–698.
- Caruana, R., Niculescu-Mizil, A., 2006. An empirical comparison of supervised learning algorithms. In: *Proc. ICML*, 10.
- Casadei, S., Mitter, S.K., 1996. Hierarchical curve reconstruction. Part I: Bifurcation analysis and recovery of smooth curves. *Proceedings of the ECCV* 1, 199–208.
- Cassot, F., Lauwers, F., Fouard, C., Prohaska, S., Valerie, L., 2006. A novel three-dimensional computer-assisted method for a quantitative study of microvascular networks of the human cerebral cortex. *Microcirculation* 13 (1), 1–18.
- Cassot, F., Lauwers, F., Lorthois, S., Puwanarajah, P., Duvernoy, H., 2009. Scaling laws for branching vessels of human cerebral cortex. *Microcirculation* 16 (4), 331–344.
- Conti, J., "VOL3D". <<http://www.mathworks.com/matlabcentral/fileexchange/4927-vol3d-m-vol3dtool-m>> (10.06.11).
- Dasgupta, S., Kalai, A., Monteleoni, C., 2009. Analysis of perceptron-based active learning. *Journal of Machine Learning Research* 10, 281–299.
- Devor, A., Tian, P., Nishimura, N., Teng, I.C., Hillman, E.M.C., Narayanan, S.N., Ulbert, I., Boas, D.A., Kleinfeld, D., Dale, A.M., 2007. Suppressed neuronal activity and concurrent arteriolar vasoconstriction may explain negative blood oxygenation level dependent signal. *Journal of Neuroscience* 27 (16), 4452–4459.
- Fawcett, T., 2004. *ROC Graphs: Notes and Practical Considerations for Researchers*. Tech. Rep. HPL-2003-4, HP Labs.
- Frangi, A.F., Niessen, W.J., Vincken, K.L., Viergever, M.A., 1998. Multiscale vessel enhancement filtering. *LNCS* 1496 (1998), 130.
- Freund, Y., Schapire, R.E., 1997a. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55, 119–139.
- Freund, Y., Seung, H.S., Shamir, E., Tishby, N., 1997b. Selective sampling using the query by committee algorithm. *Machine Learning* 28 (2–3), 133–168.
- Friman, O., Hindennach, M., Kühnel, C., Peitgen, H.-O., 2010. Multiple hypothesis template tracking of small 3d vessel structures. *Medical Image Analysis* 14 (2), 160–171.
- Fukuda, M., Moon, C.-H., Wang, P., Kim, S.-G., 2006. Mapping isoorientation columns by contrast agent-enhanced functional magnetic resonance imaging: reproducibility, specificity, and evaluation by optical imaging of intrinsic signal. *Journal of Neuroscience* 26 (46), 11821–11832.

- Gillette, T., Brown, K., Svoboda, K., Liu, Y., Ascoli, G.A., 2011. DIADeMchallenge.Org: a compendium of resources fostering the continuous development of automated neuronal reconstruction. *Neuroinformatics* 9, 303–304.
- Gonzalez, R.C., Woods, R.E., 2002. *Digital Image Processing*, second ed. Prentice-Hall, Englewood Cliffs, NJ.
- Grady, L., 2010. Method for Tracking Blood Vessels. US Patent 7711168B2, May 4, 2010.
- Grinvald, A., Sloviter, H., Vanzetta, I., 2000. Non-invasive visualization of cortical columns by fMRI. *Nature Neuroscience* 3 (2), 105–107.
- Heinzer, S., Kuhn, G., Krucker, T., Meyer, E., Ulmann-Schuler, A., Stampanoni, M., Gassmann, M., Marti, H.H., Muller, R., Vogel, J., 2008. Novel three-dimensional analysis tool for vascular trees indicates complete micro-networks, not single capillaries, as the angiogenic endpoint in mice overexpressing human VEGF(165) in the brain. *Neuroimage* 39 (4), 1549–1558.
- Helmstaedter, M., Briggman, K.L., Denk, W., 2008. 3D structural imaging of the brain with photons and electrons. *Current Opinion in Neurobiology* 18 (6), 633–641.
- Helmstaedter, M., Briggman, K.L., Denk, W., 2011. High-accuracy neurite reconstruction for high-throughput neuroanatomy. *Nature Neuroscience* 14, 1081–1088.
- Jain, V., Bollmann, B., Richardson, M., Berger, D., Helmstaedter, M., Briggman, K., Denk, W., Bowden, J., Mendenhall, J., Abraham, W., Harris, K., Kasthuri, N., Hayworth, K., Schalek, R., Tapia, J., Lichtman, J., Seung, H.S., 2010a. Boundary learning by optimization with topological constraints. *CVPR*, 2488–2495.
- Jain, V., Seung, S., Turaga, S., 2010b. Machines that learn to segment images: a crucial technology for connectomics. *Current Opinion in Neurobiology* 20, 1–14.
- Jain, V., Turaga, S.C., Briggman, K.L., Helmstaedter, M., Denk, W., Seung, H.S., 2011. Learning to Agglomerate Superpixel Hierarchies. In: *NIPS*, 10.
- Kaufhold, J.P., Hoogs, A.J., 2004. Learning to segment images using region-based perceptual features. *IEEE Proceedings of the CVPR* 2, 954–961.
- Kaufhold, J., Tsai, P.S., Blinder, P., Kleinfeld, D., 2008. Threshold relaxation is an effective means to connect gaps in 3D images of complex microvascular networks. *Workshop on Microscopic Image Analysis with Applications in Biology*. MICCAI, New York.
- Kirbas, C., Quek, F., 2004. A review of vessel extraction techniques and algorithms. *ACM Computing Surveys* 36 (2), 81–121.
- Kleinfeld, D., Bharioke, A., Blinder, P., Bock, D.D., Briggman, K.L., Chklovskii, D.B., Denk, W., Helmstaedter, M., Kaufhold, J.P., Lee, W.-C.A., Meyer, H.S., Micheva, K.D., Oberlaender, M., Prohaska, S., Reid, R.C., Smith, S.J., Takemura, S., Tsai, P.S., Sakmann, B., 2011. Large-scale automated histology in the pursuit of connectomes. *Journal of Neuroscience* 31, 16125–16138.
- Kohavi, R., 1995. A Study of Cross-Validation and Bootstrap Accuracy Estimation and Model Selection. In: *Proc. IJCAI*. <<http://robotics.stanford.edu/~ronnyk/accEst.ps>> (10.06.11).
- Krissian, K., Malandain, G., Ayache, N., Vaillant, R., Troussier, Y., 1998. Model-based multiscale detection of 3d vessels. *IEEE Proceedings of the CVPR*, 722.
- Lesage, D., Angelini, E.D., Bloch, I., Funka-Lea, G., 2009. A review of 3D vessel lumen segmentation techniques: models, features and extraction schemes. *Medical Image Analysis* 13 (6), 819–845.
- Leymarie, F.F., Giblin, P.J., Kimia, B.B., 2004. Towards surface regularization via medial axis transitions. *IEEE Proceedings of the ICPR* 3, 123–126.
- Logothetis, N.K., Pauls, J., Augath, M., Trinath, T., Oeltermann, A., 2001. Neurophysiological investigation of the basis of the fMRI signal. *Nature* 412 (6843), 150–157.
- Lu, L., Bi, J., Yu, S., Peng, Z., Krishnan, A., Zhou, X.S., 2009. Hierarchical learning for tubular structure parsing in medical imaging: a study on coronary arteries using 3D CT angiography. *Proceedings of the ICCV*, 2021–2028.
- Maddah, M., Soltanian-Zadeh, H., Afzali-Kusha, A., Shahrokni, A., Zhang, Z.G., 2005. Three-dimensional analysis of complex branching vessels in confocal microscopy images. *CMIG* 29 (6), 487–498.
- Manniesing, R., Velthuis, B.K., van Leeuwen, M.S., van der Schaaf, I.C., van Laar, P.J., Niessen, W.J., 2006a. Level set based cerebral vasculature segmentation and diameter quantification in CT angiography. *Medical Image Analysis* 10 (2), 200–214.
- Manniesing, R., Viergever, M., Niessen, W., 2006b. Vessel enhancing diffusion a scale space representation of vessel structures. *Medical Image Analysis* 10 (6), 815–825. MATLAB version 7.9.0.529 (R2009b). Natick, Massachusetts: The MathWorks Inc., 2009.
- McIntosh, C., Hamarneh, G., 2006. Vessel Crawlers: 3D Physically-based Deformable Organisms for Vasculature Segmentation and Analysis. *CVPR*, pp. 1084–1091.
- Peng, H., Long, F., Zhao, T., Myers, E., 2011. Proof-editing is the bottleneck of 3D neuron reconstruction: the problem and solutions. *Neuroinformatics*.
- Pock, T., Janko, C., Beichel, R., Bischof, H., 2005. Multiscale medialness for robust segmentation of 3D tubular structures. In: *Proc. Comp. Vis. Winter Workshop, Austria*.
- Poullis, C., You, S., Neumann, U., 2008. A Vision-based System for Automatic Detection and Extraction of Road Networks. *WACV*.
- Preibisch, S., Saalfeld, S., Tomancak, P., 2009. Globally optimal stitching of tiled 3D microscopic image acquisitions. *Bioinformatics* 25 (11), 1463–1465.
- Quinlan, J.R., 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo.
- Ren, X., Fowlkes, C.C., Malik, J., 2007. Learning probabilistic models for contour completion in natural images. *IJCV* 77, 47–63.
- Risser, L., Plouraboue, F., Descombes, X., 2007a. Gap filling in 3d vessel like patterns with tensor fields. *IEEE Proceedings of the ICCV*.
- Risser, L., Plouraboue, F., Descombes, X., 2008. Gap filling of 3-D microvascular networks by tensor voting. *IEEE Transactions on Medical Imaging* 27 (5), 674–687.
- Risser, L., Plouraboue, F., Steyer, A., Cloetens, P., Le Duc, G., 2007b. From homogeneous to fractal normal and tumorous microvascular networks in the brain. *Journal of Cerebral Blood Flow and Metabolism* 27, 293–303.
- Rothwell, C., Mundy, J., Hoffman, B., Nguyen, V.-D., 1994. *Driving Vision by Topology*. INRIA TR No. 2444.
- Schaffer, C.B., Friedman, B., Nishimura, N., Schroeder, L.F., Tsai, P.S., Ebner, F.F., Lyden, P.D., Kleinfeld, D., 2006. Two-photon imaging of cortical surface microvessels reveals a robust redistribution in blood flow after vascular occlusion. *PLoS Biology* 4 (2), 0001–0013.
- Sharon, E., Brandt, A., Basri, R., 2000. Completion energies and scale. *PAMI* 22 (10), 1117–1131.
- Szymczak, A., Tannenbaum, A., Mischaikow, K., 2005. Coronary vessel cores from 3d imagery: a topological approach. *SPIE Medical Imaging* 5747, 505–513.
- Tsai, P.S., Friedman, B., Ifarraguerri, A., Thompson, B.D., Lev-Ram, V., Schaffer, C.B., Xiong, Q., Tsien, R.Y., Squier, J.A., Kleinfeld, D., 2003. All-optical histology using ultrashort laser pulses. *Neuron* 39, 27–41.
- Tsai, P.S., Kaufhold, J.P., 2009a. VIDA\_suite 3.2 Matlab-based Routines for the Analysis of 3-D Anatomical Data Sets. <<http://physics.ucsd.edu/neurophysics/links.html>> (10.06.11).
- Tsai, P.S., Kaufhold, J.P., Blinder, P., Friedman, B., Drew, P.J., Karten, H.J., Lyden, P.D., Kleinfeld, D., 2009b. Correlations of neuronal and microvascular densities in murine cortex revealed by direct counting and colocalization of nuclei and vessels. *Journal of Neuroscience* 29 (46), 14553–14570.
- Turaga, S.C., Briggman, K.L., Helmstaedter, M., Denk, W., Seung, H.S., 2009. Maximin affinity learning of image segmentation. In: *NIPS*, 10.
- Tyrrell, J.A., Roysam, B., diTomaso, E., Tong, R., Brown, E.B., Jain, R.K., 2006. Robust 3-d modeling of tumor microvasculature using superellipsoids. *IEEE Proceedings of the ISBI*, 185–188.
- Weber, B., Keller, A.L., Reichold, J., Logothetis, N.K., 2008. The microvascular system of the striate and extrastriate visual cortex of the macaque. *Cerebral Cortex* 18 (10), 2318–2330.